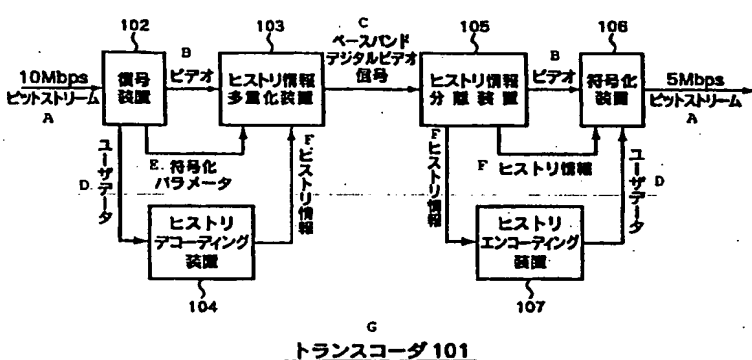


PCT

世界知的所有権機関
国際事務局

特許協力条約に基づいて公開された国際出願

(51) 国際特許分類7 H04N 7/50	A1	(11) 国際公開番号 WO00/48402 (43) 国際公開日 2000年8月17日(17.08.00)
(21) 国際出願番号 PCT/JP00/00720 (22) 国際出願日 2000年2月9日(09.02.00) (39) 優先権データ 特願平11/31944 1999年2月9日(09.02.99) JP (71) 出願人 (米国を除くすべての指定国について) ソニー株式会社(SONY CORPORATION)[JP/JP] 〒141-0001 東京都品川区北品川6丁目7番35号 Tokyo, (JP) (72) 発明者 ; および (75) 発明者 / 出願人 (米国についてのみ) 北村卓也(KITAMURA, Takuya)[JP/JP] 〒141-0001 東京都品川区北品川6丁目7番35号 ソニー株式会社内 Tokyo, (JP) (74) 代理人 弁理士 田辺恵基(TANABE, Shigemoto) 〒150-0001 東京都渋谷区神宮前1丁目11番11-508号 グリーンフアンタジアビル5階 Tokyo, (JP)		(81) 指定国 CN, JP, KR, US, 欧州特許 (DE, ES, FR, GB, NL) 添付公開書類 国際調査報告書
(54)Title: CODING SYSTEM AND ITS METHOD, CODING DEVICE AND ITS METHOD, DECODING DEVICE AND ITS METHOD, RECORDING DEVICE AND ITS METHOD, AND REPRODUCING DEVICE AND ITS METHOD (54)発明の名称 コーディングシステム及び方法、符号化装置および方法、復号化装置及び方法、記録装置及び方法、ならびに再生装置及び方法 (57) Abstract A transcoder for creating a re-coded stream including different GOPs (Groups of Pictures) structure and bit rates by re-coding a coded stream created according to the MPEG standards. Specifically, a decoding device of the transcoder creates decoded video data by decoding a source coded stream, extracts the past encoding parameters superposed in the coded stream. A coding device receives the decoded video data and the past encoding parameters and encodes the decoded video data by using the past encoding parameters. The coding device selects the encoding parameter most suitable for the application at the succeeding stage from among the past encoding parameters, and describes it in the coded stream.  <p style="text-align: center;">G トランスコーダ 101</p> <p>A...BIT STREAM B...VIDEO C...BASEBAND DIGITAL VIDEO SIGNAL D...USER DATA E...ENCODING PARAMETER F...HISTORY INFORMATION G...TRANSCODER 101 104...HISTORY DECODING DEVICE 107...HISTORY ENCODING DEVICE 102...DECODING DEVICE 103...HISTORY INFORMATION MULTIPLEXER 105...HISTORY INFORMATION SEPARATOR 106...CODING DEVICE</p>		

(57)要約

本発明は、MPEG規格に基づいて生成された符号化ストリームに対して再符号化処理を施すことによって、異なるGOP (Group of Picture) 構造や異なるビットレートを有した再符号化ストリームを生成するためのトランスコードに関する発明である。

具体的には、トランスコードの復号化装置は、ソース符号化ストリームをデコードして復号化ビデオデータを生成するとともに、符号化ストリーム中に重畳されている過去の符号化パラメータを抽出する。符号化装置は、復号化ビデオデータと過去の符号化パラメータを受け取り、過去の符号化パラメータを使用して符号化処理を行う。さらに、符号化装置は、過去の符号化パラメータの中から後段のアプリケーションに最適な符号化パラメータを選択して、符号化ストリーム中に記述する。

PCTに基づいて公開される国際出願のパンフレット第一頁に掲載されたPCT加盟国を同定するために使用されるコード(参考情報)

AE	アラブ首長国連邦	DM	ドミニカ	KZ	カザフスタン	RU	ロシア
AG	アンティグア・バーブーダ	DZ	アルジェリア	LC	セントルシア	SD	スーダン
AL	アルバニア	EE	エストニア	LI	リヒテンシュタイン	SE	スウェーデン
AM	アルメニア	ES	スペイン	LK	スリ・ランカ	SG	シンガポール
AT	オーストリア	FI	フィンランド	LR	リベリア	SI	スロヴェニア
AU	オーストラリア	FR	フランス	LS	レソト	SK	スロヴァキア
AZ	アゼルバイジャン	GA	ガボン	LT	リトアニア	SL	シエラ・レオネ
BA	ボスニア・ヘルツェゴビナ	GB	英国	LU	ルクセンブルグ	SN	セネガル
BB	バルバドス	GD	グレナダ	LV	ラトヴィア	SZ	スワジランド
BE	ベルギー	GE	グルジア	MA	モロッコ	TD	チャード
BF	ブルキナ・ファソ	GH	ガーナ	MC	モナコ	TG	トーゴ
BG	ブルガリア	GM	ガンビア	MD	モルドヴァ	TJ	タジキスタン
BJ	ベナン	GN	ギニア	MG	マダガスカル	TM	トルクメニスタン
BR	ブラジル	GR	ギリシャ	MK	マケドニア旧ユーゴスラヴィア	TR	トルコ
BY	ベラルーシ	GW	ギニア・ビサウ		共和国	TT	トリニダード・トバゴ
CA	カナダ	HR	クロアチア	ML	マリ	TZ	タンザニア
CC	中央アフリカ	HU	ハンガリー	MN	モンゴル	UA	ウクライナ
CG	コンゴ	IE	アイルランド	MR	モーリタニア	UG	ウガンダ
CH	スイス	IL	イスラエル	MW	マラウイ	US	米国
CI	コートジボワール	IN	インド	MX	メキシコ	UZ	ウズベキスタン
CM	カメルーン	IS	アイスランド	MZ	モザンビーク	VN	ヴェトナム
CN	中国	IT	イタリア	NE	ニジェール	YU	ユーゴスラヴィア
CR	コスタ・リカ	JP	日本	NL	オランダ	ZA	南アフリカ共和国
CU	キューバ	KE	ケニア	NO	ノルウェー	ZW	ジンバブエ
CY	キプロス	KG	キルギスタン	NZ	ニュージーランド		
CZ	チェコ	KP	北朝鮮	PL	ポーランド		
DE	ドイツ	KR	韓国	PT	ポルトガル		
DK	デンマーク			RO	ルーマニア		

明 細 書

コーディングシステム及び方法、符号化装置および方法、復号化装置及び方法、記録装置及び方法、ならびに再生装置及び方法

技術分野

本発明は、コーディングシステム及び方法、符号化装置および方法、復号化装置及び方法、記録装置及び方法、ならびに再生装置及び方法に関し、特に、MPEG規格に基づいて過去に符号化処理が施されたことがある符号化ストリームに対して再符号化処理を施すことによって、異なる GOP (Group of Pictures) 構造や異なるビットレートを有した再符号化ストリームを生成するためのトランスコーダに用いて好適なコーディングシステム及び方法、符号化装置および方法、復号化装置及び方法、記録装置及び方法、ならびに再生装置及び方法に関する。

背景技術

近年、テレビジョンプログラムを制作及び放送する放送局においては、ビデオデータを圧縮/符号化処理するために、MPEG (Moving Picture Experts Group) 技術が一般的に使われるようになってきた。特に、ビデオデータをテープなどのランダムアクセス可能な記録媒体素材に記録する場合、及びビデオデータをケーブルや衛星を介して伝送する場合には、この MPEG 技術がデファクトスタンダードになりつつある。

放送局において制作されたビデオプログラムが各家庭に伝送されるまでの放送局における処理の一例を簡単に説明する。まず、ビデオカメラと VTR (Video Tape Recorder) が一体となったカムコーダに設けられたエンコーダによって、ソースビデオデータをエンコード処理して磁気テープ上に記録する。この際、カムコーダのエンコーダは、VTR のテープの記録フォーマットに適するように、ソースビデオデータを符号化する。たとえば、この磁気テープ上に記録される

MPEG ビットストリームの GOP 構造は、2 フレームから 1 GOP が構成される構造（たとえば、I, B, I, B, I, B, ……）とされる。また磁気テープ上に記録されている MPEG ビットストリームのビットレートは、18 Mbps である。

次に、メイン放送局において、この磁気テープ上に記録されたビデオビットストリームを編集する編集処理を行う。そのために、磁気テープ上に記録されたビデオビットストリームの GOP 構造を、編集処理に適した GOP 構造に変換する。編集処理に適した GOP 構造とは、1 GOP が 1 フレームから構成され、すべてのピクチャが I ピクチャである GOP 構造である。なぜなら、フレーム単位で編集を行うためには、他のピクチャと相関のない I ピクチャがもっとも適しているからである。実際のオペレーションとしては、磁気テープ上に記録されたビデオストリームを一旦デコードしてベースバンドのビデオデータに戻す。そして、そのベースバンドのビデオ信号を、すべてのピクチャが I ピクチャとなるように再エンコードする。このようにデコード処理及び再エンコード処理を行うことによって、編集処理に適した GOP 構造を有したビットストリームを生成することができる。

次に、上述した編集処理によって生成された編集ビデオプログラムを、メイン局から地方局に伝送するために、編集ビデオプログラムのビットストリームを、伝送処理に適した GOP 構造及びビットレートに変換する。放送局間の伝送に適した GOP 構造とは、たとえば、1 GOP が 15 フレームから構成されている GOP 構造（たとえば、I, B, B, P, B, B, P ……）である。また、放送局間の伝送に適したビットレートは、一般的に放送局間においては、光ファイバなどの高伝送容量を有した専用線が設けられているので、50 Mbps 以上のハイビットレートであることが望ましい。具体的には、編集処理されたビデオプログラムのビットストリームを一旦デコードしてベースバンドのビデオデータに戻す。そして、そのベースバンドのビデオデータを上述した放送局間の伝送に適した GOP 構造及びビットレートを有するように再エンコードする。

地方局においては、メイン局から伝送されてきたビデオプログラムの中に、地方特有のコマーシャルを挿入するために編集処理が行われる。つまり、上述した

編集処理と同じように、メイン局から伝送されてきたビデオストリームを一旦デコードしてベースバンドのビデオデータに戻す。そして、そのベースバンドのビデオ信号を、すべてのピクチャがIピクチャとなるように再エンコードすることによって、編集処理に適したGOP構造を有したビットストリームを生成することができる。

続いて、この地方局において編集処理が行われたビデオプログラムを各家庭に、ケーブルや衛星を介して伝送するために、この伝送処理に適したGOP構造及びビットレートに変換する。たとえば、各家庭に伝送するための伝送処理に適したGOP構造とは、1GOPが15フレームから構成されるGOP構造（たとえば、I, B, B, P, B, B, P...）であって、各家庭に伝送するための伝送処理に適したビットレートは、5Mbps程度の低ビットレートである。具体的には、編集処理されたビデオプログラムのビットストリームを一旦デコードしてベースバンドのビデオデータに戻す。そして、そのベースバンドのビデオデータを上述した伝送処理に適したGOP構造及びビットレートを有するように再エンコードする。

以上の説明からも理解できるように、放送局から各家庭にビデオプログラムが伝送される間に、複数回の復号処理及び符号化処理が繰り返されている。実際には、放送局における処理は上述した信号処理以外にもさまざまな信号処理が必要であり、そのたびに復号処理及び符号化処理を繰り返さなければならない。

しかしながら、MPEG規格に基づく符号化処理及び復号処理は、100%可逆の処理ではないことは良く知られている。つまり、エンコードされる前のベースバンドのビデオデータと、デコードされた後のビデオデータは100%同じでは無く、この符号化処理及び復号処理によって画質が劣化している。つまり、上述したように、デコード処理及びエンコード処理を繰り返すと、その処理の度に、画質が劣化してしまうと言う問題があった。別の言葉で表現すると、デコード/エンコード処理を繰り返す毎に、画質の劣化が蓄積されてしまう。

本発明は、このような状況に鑑みてなされたものであり、MPEG規格に基づいて符号化された符号化ビットストリームのGOP (Group of Pictures) の構造を

変更するために復号及び符号化処理を繰り返したとしても画質劣化の発生しない
トランスコーディングシステムを実現できるようにするものである。

発明の開示

本発明は、MPEG 規格に基づいて生成された符号化ストリームに対して再符号化処理を施すことによって、異なる GOP (Group of Pictures) 構造や異なるビットレートを有した再符号化ストリームを生成するためのトランスコードに関する発明である。

具体的には、トランスコード 106 の復号化装置は、ソース符号化ストリームをデコードして復号化ビデオデータを生成するとともに、符号化ストリーム中に `history_stream()` として重畳されている過去の符号化パラメータを抽出する。この際、復号化装置は、符号化ストリーム中に `re_coding_stream_info()` として重畳されている情報に基づいて、過去の符号化パラメータを抽出する。

符号化装置は、復号化ビデオデータと過去の符号化パラメータを受け取り、過去の符号化パラメータを使用して、再符号化処理によって画質が劣化しないように、符号化処理を行い、再符号化ストリームを生成する。さらに、符号化装置は、過去の符号化パラメータの中から、この符号化装置の後段に接続されるアプリケーションに最適な符号化パラメータを選択し、選択した過去の符号化パラメータのみを `history_stream()` として符号化ストリーム中に記述する。符号化装置は、後段のアプリケーションにおいて再符号化ストリームから `history_stream()` に関する符号化パラメータを的確に抽出することができるように、選択した過去の符号化パラメータを示す情報を、`re_coding_stream_info()` として重畳する。

本発明のトランスコードによれば、後段のアプリケーションに適応した最小限の符号化パラメータで、ビデオデータの再符号化処理を繰り返したとしても、画質劣化を最小限に留めることができるコーディングシステムを実現することができる。

本発明のトランスコードは、ソース符号化ストリームを復号してビデオデータを生成するとともに、ソース符号化ストリームから過去の符号化処理により生成された過去の符号化パラメータを抽出する復号手段と、ビデオデータを再符号化し、再符号化ビデオストリームを生成する符号化手段と、過去の符号化パラメータを受け取り、過去の符号化パラメータに基づいて、符号化手段の再符号化処理を制御するとともに、過去の符号化パラメータを再符号化ストリーム中に選択的に記述する制御手段とを備えている。

本発明のトランスコードにおける符号化装置は、過去の符号化パラメータから、符号化手段の後段に接続されるアプリケーションにおいて必要となる符号化パラメータを選択し、該選択された過去の符号化パラメータを、再符号化ストリーム中に記述する。

本発明のトランスコードにおける符号化装置は、過去の符号化パラメータを選択的に再符号化ストリーム中に記述するとともに、再符号化ストリーム中に記述される過去の符号化パラメータのデータセットを示すフラグ又は／及びインジケータを、再符号化ストリーム中に記述する。

本発明のトランスコードにおける符号化装置は、過去の符号化パラメータに関する情報を、符号化ストリーム中に `history_stream()` として記述するとともに、再符号化ストリームに関する情報を、`re_coding_stream_info()` として再符号化ストリーム中に記述する。

本発明のトランスコードにおける符号化装置は、過去の符号化パラメータを選択的に再符号化ストリーム中に `history_stream()` として記述するとともに、再符号化ストリーム中に記述される過去の符号化パラメータのデータセットに関する情報を、`re_coding_stream_info()` として再符号化ストリーム中に記述する。

本発明の符号化装置は、ビデオデータに関して施された過去の符号化処理に関する過去の符号化パラメータを受け取り、過去の符号化パラメータを選択的に再符号化ストリーム中に記述するとともに、再符号化ストリーム中に記述される過去の符号化パラメータのデータセットを示す情報を、再符号化ストリーム中に記

述する。

本発明の復号化装置は、符号化ストリームから、符号化ストリーム中に重畳されている過去の符号化パラメータのデータセットに関する情報を抽出し、このデータセットに関する情報に基づいて、符号化ストリームから、過去の符号化パラメータを抽出する。

本発明の復号化装置は、符号化ストリームから、符号化ストリーム中に `re_coding_stream_info()` として記述されているフラグ又は／及びインジケータを抽出し、このフラグ又は／及びインジケータに基づいて、符号化ストリームから、過去の符号化パラメータを抽出する。

図面の簡単な説明

図 1 は、高効率符号化の原理を説明する図である。

図 2 は、画像データを圧縮する場合におけるピクチャタイプを説明する図である。

図 3 は、画像データを圧縮する場合におけるピクチャタイプを説明する図である。

図 4 は、動画像信号を符号化する原理を説明する図である。

図 5 は、動画像信号を符号化し、復号する装置の構成を示すブロック図である。

図 6 は、画像データの構成を説明する図である。

図 7 は、図 5 のエンコーダ 18 の構成を示すブロック図である。

図 8 は、図 7 の予測モード切替回路 52 の動作を説明する図である。

図 9 は、図 7 の予測モード切替回路 52 の動作を説明する図である。

図 10 は、図 7 の予測モード切替回路 52 の動作を説明する図である。

図 11 は、図 7 の予測モード切替回路 52 の動作を説明する図である。

図 12 は、図 5 のデコーダ 31 の構成を示すブロック図である。

図 13 は、ピクチャタイプに対応した SNR 制御を説明する図である。

図14は、本発明を適用したトランスコーダ101の構成を示すブロック図である。

図15は、図14のトランスコーダ101のより詳細な構成を示すブロック図である。

図16は、図14の復号装置102に内蔵される復号化装置102の構成を示すブロック図である。

図17は、マクロブロックの画素を説明する図である。

図18は、符号化パラメータが記録される領域を説明する図である。

図19は、図14の符号化装置106に内蔵される符号化装置106の構成を示すブロック図である。

図20は、図15のヒストリ VLC 211の構成例を示すブロック図である。

図21は、図15のヒストリ VLD 203の構成例を示すブロック図である。

図22は、図15のコンバータ 212の構成例を示すブロック図である。

図23は、図22のスタッフ回路 323の構成例を示すブロック図である。

図24は、図22のコンバータ 212の動作を説明するタイミングチャートである。

図25は、図15のコンバータ 202の構成例を示すブロック図である。

図26は、図25のディリート回路 343の構成例を示すブロック図である。

図27は、図15のコンバータ 212の他の構成例を示すブロック図である。

図28は、図15のコンバータ 202の他の構成例を示すブロック図である。

図29は、図15のユーザデータフォーマッタ 213の構成例を示すブロック図である。

図30は、図14のトランスコーダ101が実際に使用される状態を示す図である。

図31は、符号化パラメータが記録される領域を説明する図である。

図32は、図14の符号化装置106の変更可能ピクチャタイプ判定処理を説明するフローチャートである。

図 3 3 は、ピクチャタイプが変更される例を示す図である。

図 3 4 は、ピクチャタイプが変更される他の例を示す図である。

図 3 5 は、図 1 4 の符号化装置 1 0 6 の量子化制御処理を説明する図である。

図 3 6 は、図 1 4 の符号化装置 1 0 6 の量子化制御処理を説明するフローチャートである。

図 3 7 は、密結合されたトランスコード 1 0 1 の構成を示すブロック図である。

図 3 8 は、ビデオシーケンスのストリームのシンタックスを説明する図である。

図 3 9 は、図 3 8 のシンタックスの構成を説明する図である。

図 4 0 は、固定長の履歴情報を記録する `history_stream()` のシンタックスを説明する図である。

図 4 1 は、固定長の履歴情報を記録する `history_stream()` のシンタックスを説明する図である。

図 4 2 は、固定長の履歴情報を記録する `history_stream()` のシンタックスを説明する図である。

図 4 3 は、固定長の履歴情報を記録する `history_stream()` のシンタックスを説明する図である。

図 4 4 は、固定長の履歴情報を記録する `history_stream()` のシンタックスを説明する図である。

図 4 5 は、固定長の履歴情報を記録する `history_stream()` のシンタックスを説明する図である。

図 4 6 は、固定長の履歴情報を記録する `history_stream()` のシンタックスを説明する図である。

図 4 7 は、可変長の履歴情報を記録する `history_stream()` のシンタックスを説明する図である。

図 4 8 は、`sequence_header()` のシンタックスを説明する図である。

図 4 9 は、sequence_extension() のシンタックスを説明する図である。

図 5 0 は、extension_and_user_data() のシンタックスを説明する図である。

図 5 1 は、user_data() のシンタックスを説明する図である。

図 5 2 は、group_of_pictures_header() のシンタックスを説明する図である。

図 5 3 は、picture_header() のシンタックスを説明する図である。

図 5 4 は、picture_coding_extension() のシンタックスを説明する図である。

図 5 5 は、extension_data() のシンタックスを説明する図である。

図 5 6 は、quant_matrix_extension() のシンタックスを説明する図である。

図 5 7 は、copyright_extension() のシンタックスを説明する図である。

図 5 8 は、picture_display_extension() のシンタックスを説明する図である。

図 5 9 は、picture_data() のシンタックスを説明する図である。

図 6 0 は、slice() のシンタックスを説明する図である。

図 6 1 は、macroblock() のシンタックスを説明する図である。

図 6 2 は、macroblock_modes() のシンタックスを説明する図である。

図 6 3 は、motion_vectors(s) のシンタックスを説明する図である。

図 6 4 は、motion_vector(r,s) のシンタックスを説明する図である。

図 6 5 は、I ピクチャに対する macroblock_type の可変長符号を説明する図である。

図 6 6 は、P ピクチャに対する macroblock_type の可変長符号を説明する図である。

図 6 7 は、B ピクチャに対する macroblock_type の可変長符号を説明する図である。

図 6 8 は、re_coding_stream_info() のシンタックスを説明する図である。

図 6 9 は、red_bw_flag, red_bw_indicator を説明する図である。

図 7 0 は、履歴情報のコーディングパラメータのデータセットを説明する図である。

図 7 1 は、Re_Coding Information Bus macroblock formation を説明する図である。

図 7 2 は、Picture rate elements を説明する図である。

図 7 3 は、Picture rate elements を説明する図である。

図 7 4 は、Picture rate elements を説明する図である。

図 7 5 は、Re_Coding Information Bus が記録される領域を説明する図である。

図 7 6 は、ビデオテープレコーダの記録系の構成例を表すブロック図である。

図 7 7 は、ビデオテープレコーダの再生系の構成例を表すブロック図である。

図 7 8 は、ビデオテープレコーダの記録系の他の構成例を表すブロック図である。

図 7 9 は、ビデオテープレコーダの再生系の他の構成例を表すブロック図である。

図 8 0 は、ビデオストリームと history_stream の記録位置を説明する図である。

発明を実施するための最良の形態

以下に、本発明を適用したトランスコーダについて説明するが、その前に、動画像信号の圧縮符号化について説明する。なお、本明細書においてシステムの用語は、複数の装置、手段などにより構成される全体的な装置を意味するものである。

例えば、テレビ会議システム、テレビ電話システムなどのように、動画像信号を遠隔地に伝送するシステムにおいては、伝送路を効率良く利用するため、映像信号のライン相関やフレーム間相関を利用して、画像信号を圧縮符号化されるようになされている。

ライン相関を利用すると、画像信号を、例えば DCT（離散コサイン変換）処理するなどして圧縮することができる。

また、フレーム間相関を利用すると、画像信号をさらに圧縮して符号化することが可能となる。例えば図1に示すように、時刻 t_1 乃至 t_3 において、フレーム画像 PC_1 乃至 PC_3 がそれぞれ発生している場合、フレーム画像 PC_1 および PC_2 の画像信号の差を演算して、 PC_{12} を生成し、また、フレーム画像 PC_2 および PC_3 の差を演算して、 PC_{23} を生成する。通常、時間的に隣接するフレームの画像は、それ程大きな変化を有していないため、両者の差を演算すると、その差分信号は小さな値のものとなる。そこで、この差分信号を符号化すれば、符号量を圧縮することができる。

しかしながら、差分信号のみを伝送したのでは、元の画像を復元することができない。そこで、各フレームの画像を、Iピクチャ、PピクチャまたはBピクチャの3種類のピクチャタイプのいずれかとし、画像信号を圧縮符号化するようにしている。

すなわち、例えば図2に示すように、フレーム F_1 乃至 F_{17} までの17フレームの画像信号をグループオブピクチャ(GOP)とし、処理の1単位とする。そして、その先頭のフレーム F_1 の画像信号はIピクチャとして符号化し、第2番目のフレーム F_2 はBピクチャとして、また第3番目のフレーム F_3 はPピクチャとして、それぞれ処理する。以下、第4番目以降のフレーム F_4 乃至 F_{17} は、BピクチャまたはPピクチャとして交互に処理する。

Iピクチャの画像信号としては、その1フレーム分の画像信号をそのまま伝送する。これに対して、Pピクチャの画像信号としては、基本的には、図2に示すように、それより時間的に先行するIピクチャまたはPピクチャの画像信号からの差分を伝送する。さらにBピクチャの画像信号としては、基本的には、図3に示すように、時間的に先行するフレームまたは後行するフレームの両方の平均値からの差分を求め、その差分を符号化する。

図4は、このようにして、動画像信号を符号化する方法の原理を示している。同図に示すように、最初のフレーム F_1 は、Iピクチャとして処理されるため、そのまま伝送データ F_1X として伝送路に伝送される(画像内符号化)。これに

対して、第2のフレームF2は、Bピクチャとして処理されるため、時間的に先行するフレームF1と、時間的に後行するフレームF3の平均値との差分が演算され、その差分が伝送データF2Xとして伝送される。

ただし、このBピクチャとしての処理は、さらに細かく説明すると、4種類存在する。その第1の処理は、元のフレームF2のデータをそのまま伝送データF2Xとして伝送するものであり（SP1）（イントラ符号化）、Iピクチャにおける場合と同様の処理となる。第2の処理は、時間的に後のフレームF3からの差分を演算し、その差分（SP2）を伝送するものである（後方予測符号化）。第3の処理は、時間的に先行するフレームF1との差分（SP3）を伝送するものである（前方予測符号化）。さらに第4の処理は、時間的に先行するフレームF1と後行するフレームF3の平均値との差分（SP4）を生成し、これを伝送データF2Xとして伝送するものである（両方向予測符号化）。

実際には、上述した4つの方法のうちの伝送データが最も少なくなる方法が採用される。

なお、差分データを伝送するとき、差分を演算する対象となるフレームの画像（参照画像）との間の動きベクトルx1（フレームF1とF2の間の動きベクトル）（前方予測の場合）、もしくはx2（フレームF3とF2の間の動きベクトル）（後方予測の場合）、またはx1とx2の両方（両方向予測の場合）が、差分データとともに伝送される。

また、PピクチャのフレームF3は、時間的に先行するフレームF1を参照画像として、このフレームとの差分信号（SP3）と、動きベクトルx3が演算され、これが伝送データF3Xとして伝送される（前方予測符号化）。あるいはまた、元のフレームF3のデータが、そのままデータF3Xとして伝送される（SP1）（イントラ符号化）。これらの方法のうち、Bピクチャにおける場合と同様に、伝送データがより少なくなる方法が選択される。

図5は、上述した原理に基づいて、動画像信号を符号化して伝送し、これを復号する装置の構成例を示している。符号化装置1は、入力された映像信号を符号

化し、伝送路としての記録媒体 3 に伝送するようになされている。そして、復号装置 2 は、記録媒体 3 に記録された信号を再生し、これを復号して出力するようになされている。

符号化装置 1 においては、入力された映像信号が前処理回路 1 1 に入力され、そこで輝度信号と色信号（本実施の形態の場合、色差信号）が分離され、それぞれ A/D 変換器 1 2, 1 3 でアナログ信号からデジタル信号に変換される。A/D 変換器 1 2, 1 3 によりデジタル信号に変換された映像信号は、フレームメモリ 1 4 に供給され、記憶される。フレームメモリ 1 4 は、輝度信号を輝度信号フレームメモリ 1 5 に、また、色差信号を色差信号フレームメモリ 1 6 に、それぞれ記憶させる。

フォーマット変換回路 1 7 は、フレームメモリ 1 4 に記憶されたフレームフォーマットの信号を、ブロックフォーマットの信号に変換する。すなわち、図 6 に示すように、フレームメモリ 1 4 に記憶された映像信号は、1 ライン当り H ドットのラインが V ライン集められた、図 6 (A) に示すようなフレームフォーマットのデータとされている。フォーマット変換回路 1 7 は、この 1 フレームの信号を、図 6 (B) に示すように、16 ラインを単位として M 個のスライスに区分する。そして、各スライスは、M 個のマクロブロックに分割される。マクロブロックは、図 6 (C) に示すように、16 × 16 個の画素（ドット）に対応する輝度信号により構成され、この輝度信号は、さらに 8 × 8 ドットを単位とするブロック Y [1] 乃至 Y [4] に区分される。そして、この 16 × 16 ドットの輝度信号には、8 × 8 ドットの C b 信号と、8 × 8 ドットの C r 信号が対応される。

このように、ブロックフォーマットに変換されたデータは、フォーマット変換回路 1 7 からエンコーダ 1 8 に供給され、ここでエンコード（符号化）が行われる。その詳細については、図 7 を参照して後述する。

エンコーダ 1 8 によりエンコードされた信号は、ビットストリームとして伝送路に出力される。例えば記録回路 1 9 に供給され、デジタル信号として記録媒体 3 に記録される。

復号装置 2 の再生回路 30 により記録媒体 3 より再生されたデータは、デコーダ 31 に供給され、デコードされる。デコーダ 31 の詳細については、図 12 を参照して後述する。

デコーダ 31 によりデコードされたデータは、フォーマット変換回路 32 に入力され、ブロックフォーマットからフレームフォーマットに変換される。そして、フレームフォーマットの輝度信号は、フレームメモリ 33 の輝度信号フレームメモリ 34 に供給されて記憶され、色差信号は色差信号フレームメモリ 35 に供給されて記憶される。輝度信号フレームメモリ 34 と色差信号フレームメモリ 35 から読み出された輝度信号と色差信号は、それぞれ D/A 変換器 36, 37 によりアナログ信号に変換され、後処理回路 38 に供給される。後処理回路 38 は、輝度信号と色差信号を合成して出力する。

次に図 7 を参照して、エンコーダ 18 の構成について説明する。符号化される画像データは、マクロブロック単位で動きベクトル検出回路 50 に入力される。動きベクトル検出回路 50 は、予め設定されている所定のシーケンスに従って、各フレームの画像データを、I ピクチャ、P ピクチャ、または B ピクチャとして処理する。シーケンシャルに入力される各フレームの画像を、I, P、または B のいずれのピクチャとして処理するかは、予め定められている（例えば、図 2 と図 3 に示したように、フレーム F1 乃至 F17 により構成されるグループオブピクチャが、I, B, P, B, P, . . . B, P として処理される）。

I ピクチャとして処理されるフレーム（例えば、フレーム F1）の画像データは、動きベクトル検出回路 50 からフレームメモリ 51 の前方原画像部 51a に転送、記憶され、B ピクチャとして処理されるフレーム（例えば、フレーム F2）の画像データは、原画像部 51b に転送、記憶され、P ピクチャとして処理されるフレーム（例えば、フレーム F3）の画像データは、後方原画像部 51c に転送、記憶される。

また、次のタイミングにおいて、さらに B ピクチャ（フレーム F4）または P ピクチャ（フレーム F5）として処理すべきフレームの画像が入力されたとき、

それまで後方原画像部 5 1 c に記憶されていた最初の P ピクチャ（フレーム F 3）の画像データが、前方原画像部 5 1 a に転送され、次の B ピクチャ（フレーム F 4）の画像データが、参照原画像部 5 1 b に記憶（上書き）され、次の P ピクチャ（フレーム F 5）の画像データが、後方原画像部 5 1 c に記憶（上書き）される。このような動作が順次繰り返される。

フレームメモリ 5 1 に記憶された各ピクチャの信号は、そこから読み出され、予測モード切り替え回路 5 2 において、フレーム予測モード処理、またはフィールド予測モード処理が行なわれる。

さらにまた、予測判定回路 5 4 の制御の下に、演算器 5 3 において、画像内予測、前方予測、後方予測、または両方向予測の演算が行なわれる。これらの処理のうち、いずれの処理を行なうかは、予測誤差信号（処理の対象とされている参照画像と、これに対する予測画像との差分）に対応して決定される。このため、動きベクトル検出回路 5 0 は、この判定に用いられる予測誤差信号の絶対値和（自乗和でもよい）を生成する。

ここで、予測モード切り替え回路 5 2 におけるフレーム予測モードとフィールド予測モードについて説明する。

フレーム予測モードが設定された場合においては、予測モード切り替え回路 5 2 は、動きベクトル検出回路 5 0 より供給される 4 個の輝度ブロック Y [1] 乃至 Y [4] を、そのまま後段の演算器 5 3 に出力する。すなわち、この場合においては、図 8 に示すように、各輝度ブロックに奇数フィールドのラインのデータと、偶数フィールドのラインのデータとが混在した状態となっている。このフレーム予測モードにおいては、4 個の輝度ブロック（マクロブロック）を単位として予測が行われ、4 個の輝度ブロックに対して 1 個の動きベクトルが対応される。

これに対して、予測モード切り替え回路 5 2 は、フィールド予測モードにおいては、図 8 に示す構成で動きベクトル検出回路 5 0 より入力される信号を、図 9 に示すように、4 個の輝度ブロックのうち、輝度ブロック Y [1] と Y [2] を

、例えば奇数フィールドのラインのドットだけで構成させ、他の2個の輝度ブロックY[3]とY[4]を、偶数フィールドのラインのドットだけで構成させて、演算器53に出力する。この場合においては、2個の輝度ブロックY[1]とY[2]に対して、1個の動きベクトルが対応され、他の2個の輝度ブロックY[3]とY[4]に対して、他の1個の動きベクトルが対応される。

動きベクトル検出回路50は、フレーム予測モードにおける予測誤差の絶対値和、およびフィールド予測モードにおける予測誤差の絶対値和を予測モード切り替え回路52に出力する。予測モード切り替え回路52は、フレーム予測モードとフィールド予測モードにおける予測誤差の絶対値和を比較し、その値が小さい予測モードに対応する処理を施して、データを演算器53に出力する。

ただし、このような処理は、実際には動きベクトル検出回路50で行われる。すなわち、動きベクトル検出回路50は、決定されたモードに対応する構成の信号を予測モード切り替え回路52に出力し、予測モード切り替え回路52は、その信号を、そのまま後段の演算器53に出力する。

なお、色差信号は、フレーム予測モードの場合、図8に示すように、奇数フィールドのラインのデータと偶数フィールドのラインのデータとが混在する状態で、演算器53に供給される。また、フィールド予測モードの場合、図9に示すように、各色差ブロックCb, Crの上半分(4ライン)が、輝度ブロックY[1], Y[2]に対応する奇数フィールドの色差信号とされ、下半分(4ライン)が、輝度ブロックY[3], Y[4]に対応する偶数フィールドの色差信号とされる。

また、動きベクトル検出回路50は、以下に示すようにして、予測判定回路54において、画像内予測、前方予測、後方予測、または両方向予測のいずれの予測を行なうかを決定するための予測誤差の絶対値和を生成する。

すなわち、画像内予測の予測誤差の絶対値和として、参照画像のマクロブロックの信号 A_{ij} の総和 ΣA_{ij} の絶対値 $|\Sigma A_{ij}|$ と、マクロブロックの信号 A_{ij} の絶対値 $|A_{ij}|$ の総和 $\Sigma |A_{ij}|$ の差を求める。また、前方予測の予測誤差の

絶対値和として、参照画像のマクロブロックの信号 A_{ij} と、予測画像のマクロブロックの信号 B_{ij} の差 $A_{ij}-B_{ij}$ の絶対値 $|A_{ij}-B_{ij}|$ の総和 $\sum |A_{ij}-B_{ij}|$ を求める。また、後方予測と両方向予測の予測誤差の絶対値和も、前方予測における場合と同様に（その予測画像を前方予測における場合と異なる予測画像に変更して）求める。

これらの絶対値和は、予測判定回路54に供給される。予測判定回路54は、前方予測、後方予測および両方向予測の予測誤差の絶対値和のうちの最も小さいものを、インタ予測の予測誤差の絶対値和として選択する。さらに、このインタ予測の予測誤差の絶対値和と、画像内予測の予測誤差の絶対値和とを比較し、その小さい方を選択し、この選択した絶対値和に対応するモードを予測モードとして選択する。すなわち、画像内予測の予測誤差の絶対値和の方が小さければ、画像内予測モードが設定される。インタ予測の予測誤差の絶対値和の方が小さければ、前方予測、後方予測または両方向予測モードのうちの対応する絶対値和が最も小さかったモードが設定される。

このように、動きベクトル検出回路50は、参照画像のマクロブロックの信号を、フレームまたはフィールド予測モードのうち、予測モード切り替え回路52により選択されたモードに対応する構成で、予測モード切り替え回路52を介して演算器53に供給するとともに、4つの予測モードのうちの予測判定回路54により選択された予測モードに対応する予測画像と参照画像の間の動きベクトルを検出し、可変長符号化回路58と動き補償回路64に出力する。上述したように、この動きベクトルとしては、対応する予測誤差の絶対値和が最小となるものが選択される。

予測判定回路54は、動きベクトル検出回路50が前方原画像部51aよりIピクチャの画像データを読み出しているとき、予測モードとして、フレームまたはフィールド（画像）内予測モード（動き補償を行わないモード）を設定し、演算器53のスイッチ53dを接点a側に切り替える。これにより、Iピクチャの画像データがDCTモード切り替え回路55に入力される。

DCT モード切り替え回路 55 は、図 10 または図 11 に示すように、4 個の輝度ブロックのデータを、奇数フィールドのラインと偶数フィールドのラインが混在する状態（フレーム DCT モード）、または、分離された状態（フィールド DCT モード）、のいずれかの状態にして、DCT 回路 56 に出力する。

すなわち、DCT モード切り替え回路 55 は、奇数フィールドと偶数フィールドのデータを混在して DCT 処理した場合における符号化効率と、分離した状態において DCT 処理した場合の符号化効率とを比較し、符号化効率の良好なモードを選択する。

例えば、入力された信号を、図 10 に示すように、奇数フィールドと偶数フィールドのラインが混在する構成とし、上下に隣接する奇数フィールドのラインの信号と偶数フィールドのラインの信号の差を演算し、さらにその絶対値の和（または自乗和）を求める。

また、入力された信号を、図 11 に示すように、奇数フィールドと偶数フィールドのラインが分離した構成とし、上下に隣接する奇数フィールドのライン同士の信号の差と、偶数フィールドのライン同士の信号の差を演算し、それぞれの絶対値の和（または自乗和）を求める。

さらに、両者（絶対値和）を比較し、小さい値に対応する DCT モードを設定する。すなわち、前者の方が小さければ、フレーム DCT モードを設定し、後者の方が小さければ、フィールド DCT モードを設定する。

そして、選択した DCT モードに対応する構成のデータを DCT 回路 56 に出力するとともに、選択した DCT モードを示す DCT フラグを、可変長符号化回路 58、および動き補償回路 64 に出力する。

予測モード切り替え回路 52 における予測モード（図 8 と図 9）と、この DCT モード切り替え回路 55 における DCT モード（図 10 と図 11）を比較して明らかなように、輝度ブロックに関しては、両者の各モードにおけるデータ構造は実質的に同一である。

予測モード切り替え回路 52 において、フレーム予測モード（奇数ラインと偶

数ラインが混在するモード) が選択された場合、DCT モード切り替え回路 5 5 においても、フレーム DCT モード (奇数ラインと偶数ラインが混在するモード) が選択される可能性が高く、また予測モード切り替え回路 5 2 において、フィールド予測モード (奇数フィールドと偶数フィールドのデータが分離されたモード) が選択された場合、DCT モード切り替え回路 5 5 において、フィールド DCT モード (奇数フィールドと偶数フィールドのデータが分離されたモード) が選択される可能性が高い。

しかしながら、必ずしも常にこのようにモードが選択されるわけではなく、予測モード切り替え回路 5 2 においては、予測誤差の絶対値和が小さくなるようにモードが決定され、DCT モード切り替え回路 5 5 においては、符号化効率が良好となるようにモードが決定される。

DCT モード切り替え回路 5 5 より出力された I ピクチャの画像データは、DCT 回路 5 6 に入力されて DCT 処理され、DCT 係数に変換される。この DCT 係数は、量子化回路 5 7 に入力され、送信バッファ 5 9 のデータ蓄積量 (バッファ蓄積量) に対応した量子化スケールで量子化された後、可変長符号化回路 5 8 に入力される。

可変長符号化回路 5 8 は、量子化回路 5 7 より供給される量子化スケール (スケール) に対応して、量子化回路 5 7 より供給される画像データ (いまの場合、I ピクチャのデータ) を、例えばハフマン符号などの可変長符号に変換し、送信バッファ 5 9 に出力する。

可変長符号化回路 5 8 にはまた、量子化回路 5 7 より量子化スケール (スケール)、予測判定回路 5 4 より予測モード (画像内予測、前方予測、後方予測、または両方向予測のいずれが設定されたかを示すモード)、動きベクトル検出回路 5 0 より動きベクトル、予測モード切り替え回路 5 2 より予測フラグ (フレーム予測モードまたはフィールド予測モードのいずれが設定されたかを示すフラグ)、および DCT モード切り替え回路 5 5 が出力する DCT フラグ (フレーム DCT モードまたはフィールド DCT モードのいずれが設定されたかを示すフラグ) が入力さ

れており、これらも可変長符号化される。

送信バッファ 59 は、入力されたデータを一時蓄積し、蓄積量に対応するデータを量子化回路 57 に出力する。送信バッファ 59 は、そのデータ残量が許容上限値まで増量すると、量子化制御信号によって量子化回路 57 の量子化スケールを大きくすることにより、量子化データのデータ量を低下させる。また、これとは逆に、データ残量が許容下限値まで減少すると、送信バッファ 59 は、量子化制御信号によって量子化回路 57 の量子化スケールを小さくすることにより、量子化データのデータ量を増大させる。このようにして、送信バッファ 59 のオーバフローまたはアンダフローが防止される。

そして、送信バッファ 59 に蓄積されたデータは、所定のタイミングで読み出され、伝送路に出力され、例えば記録回路 19 を介して記録媒体 3 に記録される。

一方、量子化回路 57 より出力された I ピクチャのデータは、逆量子化回路 60 に入力され、量子化回路 57 より供給される量子化スケールに対応して逆量子化される。逆量子化回路 60 の出力は、IDCT（逆離散コサイン変換）回路 61 に入力され、逆離散コサイン変換処理された後、演算器 62 を介してフレームメモリ 63 の前方予測画像部 63 a 供給されて記憶される。

動きベクトル検出回路 50 は、シーケンシャルに入力される各フレームの画像データを、たとえば、I, B, P, B, P, B... のピクチャとしてそれぞれ処理する場合、最初に入力されたフレームの画像データを I ピクチャとして処理した後、次に入力されたフレームの画像を B ピクチャとして処理する前に、さらにその次に入力されたフレームの画像データを P ピクチャとして処理する。B ピクチャは、後方予測を伴うため、後方予測画像としての P ピクチャが先に用意されていないと、復号することができないからである。

そこで動きベクトル検出回路 50 は、I ピクチャの処理の次に、後方原画像部 51 c に記憶されている P ピクチャの画像データの処理を開始する。そして、上述した場合と同様に、マクロブロック単位でのフレーム間差分（予測誤差）の絶

対値和が、動きベクトル検出回路 50 から予測モード切り替え回路 52 と予測判定回路 54 に供給される。予測モード切り替え回路 52 と予測判定回路 54 は、この P ピクチャのマクロブロックの予測誤差の絶対値和に対応して、フレーム／フィールド予測モード、または画像内予測、前方予測、後方予測、もしくは両方向予測の予測モードを設定する。

演算器 53 は、画像内予測モードが設定されたとき、スイッチ 53 d を上述したように接点 a 側に切り替える。したがって、このデータは、I ピクチャのデータと同様に、DCT モード切り替え回路 55、DCT 回路 56、量子化回路 57、可変長符号化回路 58、および送信バッファ 59 を介して伝送路に伝送される。また、このデータは、逆量子化回路 60、IDCT 回路 61、および演算器 62 を介してフレームメモリ 63 の後方予測画像部 63 b に供給されて記憶される。

また、前方予測モードが設定された場合、スイッチ 53 d が接点 b に切り替えられるとともに、フレームメモリ 63 の前方予測画像部 63 a に記憶されている画像（いまの場合、I ピクチャの画像）データが読み出され、動き補償回路 64 により、動きベクトル検出回路 50 が出力する動きベクトルに対応して動き補償される。すなわち、動き補償回路 64 は、予測判定回路 54 より前方予測モードの設定が指令されたとき、前方予測画像部 63 a の読み出しアドレスを、動きベクトル検出回路 50 が、現在、出力しているマクロブロックの位置に対応する位置から動きベクトルに対応する分だけずらしてデータを読み出し、予測画像データを生成する。

動き補償回路 64 より出力された予測画像データは、演算器 53 a に供給される。演算器 53 a は、予測モード切り替え回路 52 より供給された参照画像のマクロブロックのデータから、動き補償回路 65 より供給された、このマクロブロックに対応する予測画像データを減算し、その差分（予測誤差）を出力する。この差分データは、DCT モード切り替え回路 55、DCT 回路 56、量子化回路 57、可変長符号化回路 58、および送信バッファ 59 を介して伝送路に伝送される。また、この差分データは、逆量子化回路 60、および IDCT 回路 61 により局

所的に復号され、演算器 6 2 に入力される。

この演算器 6 2 にはまた、演算器 5 3 a に供給されている予測画像データと同一のデータが供給されている。演算器 6 2 は、IDCT 回路 6 1 が出力する差分データに、動き補償回路 6 4 が出力する予測画像データを加算する。これにより、元の（復号した）P ピクチャの画像データが得られる。この P ピクチャの画像データは、フレームメモリ 6 3 の後方予測画像部 6 3 b に供給されて記憶される。

動きベクトル検出回路 5 0 は、このように、I ピクチャと P ピクチャのデータが前方予測画像部 6 3 a と後方予測画像部 6 3 b にそれぞれ記憶された後、次に B ピクチャの処理を実行する。予測モード切り替え回路 5 2 と予測判定回路 5 4 は、マクロブロック単位でのフレーム間差分の絶対値和の大きさに対応して、フレーム／フィールドモードを設定し、また、予測モードを画像内予測モード、前方予測モード、後方予測モード、または両方向予測モードのいずれかに設定する。

上述したように、画像内予測モードまたは前方予測モードの時、スイッチ 5 3 d は接点 a または b に切り替えられる。このとき、P ピクチャにおける場合と同様の処理が行われ、データが伝送される。

これに対して、後方予測モードまたは両方向予測モードが設定された時、スイッチ 5 3 d は、接点 c または d にそれぞれ切り替えられる。

スイッチ 5 3 d が接点 c に切り替えられている後方予測モードの時、後方予測画像部 6 3 b に記憶されている画像（いまの場合、P ピクチャの画像）データが読み出され、動き補償回路 6 4 により、動きベクトル検出回路 5 0 が出力する動きベクトルに対応して動き補償される。すなわち、動き補償回路 6 4 は、予測判定回路 5 4 より後方予測モードの設定が指令されたとき、後方予測画像部 6 3 b の読み出しアドレスを、動きベクトル検出回路 5 0 が、現在、出力しているマクロブロックの位置に対応する位置から動きベクトルに対応する分だけずらしてデータを読み出し、予測画像データを生成する。

動き補償回路 6 4 より出力された予測画像データは、演算器 5 3 b に供給され

る。演算器 53b は、予測モード切り替え回路 52 より供給された参照画像のマクロブロックのデータから、動き補償回路 64 より供給された予測画像データを減算し、その差分を出力する。この差分データは、DCT モード切り替え回路 55、DCT 回路 56、量子化回路 57、可変長符号化回路 58、および送信バッファ 59 を介して伝送路に伝送される。

スイッチ 53d が接点 d に切り替えられている両方向予測モードの時、前方予測画像部 63a に記憶されている画像（いまの場合、I ピクチャの画像）データと、後方予測画像部 63b に記憶されている画像（いまの場合、P ピクチャの画像）データが読み出され、動き補償回路 64 により、動きベクトル検出回路 50 が出力する動きベクトルに対応して動き補償される。

すなわち、動き補償回路 64 は、予測判定回路 54 より両方向予測モードの設定が指令されたとき、前方予測画像部 63a と後方予測画像部 63b の読み出しアドレスを、動きベクトル検出回路 50 がいま出力しているマクロブロックの位置に対応する位置から動きベクトル（この場合の動きベクトルは、前方予測画像用と後方予測画像用の 2 つとなる）に対応する分だけずらしてデータを読み出し、予測画像データを生成する。

動き補償回路 64 より出力された予測画像データは、演算器 53c に供給される。演算器 53c は、動きベクトル検出回路 50 より供給された参照画像のマクロブロックのデータから、動き補償回路 64 より供給された予測画像データの平均値を減算し、その差分を出力する。この差分データは、DCT モード切り替え回路 55、DCT 回路 56、量子化回路 57、可変長符号化回路 58、および送信バッファ 59 を介して伝送路に伝送される。

B ピクチャの画像は、他の画像の予測画像とされることがないため、フレームメモリ 63 には記憶されない。

なお、フレームメモリ 63 において、前方予測画像部 63a と後方予測画像部 63b は、必要に応じてバンク切り替えが行われ、所定の参照画像に対して、一方または他方に記憶されているものを、前方予測画像あるいは後方予測画像とし

て切り替えて出力することができる。

上述した説明においては、輝度ブロックを中心として説明をしたが、色差ブロックについても同様に、図8乃至図11に示すマクロブロックを単位として処理されて伝送される。なお、色差ブロックを処理する場合の動きベクトルは、対応する輝度ブロックの動きベクトルを垂直方向と水平方向に、それぞれ1/2にしたものが用いられる。

図12は、図5のデコーダ31の構成を示すブロック図である。伝送路（記録媒体3）を介して伝送された符号化された画像データは、図示せぬ受信回路で受信されたり、再生装置で再生され、受信バッファ81に一時記憶された後、復号回路90の可変長復号回路82に供給される。可変長復号回路82は、受信バッファ81より供給されたデータを可変長復号し、動きベクトル、予測モード、予測フラグ、およびDCTフラグを動き補償回路87に出力し、量子化スケールを逆量子化回路83に出力するとともに、復号された画像データを逆量子化回路83に出力する。

逆量子化回路83は、可変長復号回路82より供給された画像データを、同じく可変長復号回路82より供給された量子化スケールに従って逆量子化し、IDCT回路84に出力する。逆量子化回路83より出力されたデータ（DCT係数）は、IDCT回路84により、逆離散コサイン変換処理が施され、演算器85に供給される。

IDCT回路84より演算器85に供給された画像データが、Iピクチャのデータである場合、そのデータは演算器85より出力され、演算器85に後に入力される画像データ（PまたはBピクチャのデータ）の予測画像データ生成のために、フレームメモリ86の前方予測画像部86aに供給されて記憶される。また、このデータは、フォーマット変換回路32（図5）に出力される。

IDCT回路84より供給された画像データが、その1フレーム前の画像データを予測画像データとするPピクチャのデータであり、前方予測モードのデータである場合、フレームメモリ86の前方予測画像部86aに記憶されている、1フ

フレーム前の画像データ（Iピクチャのデータ）が読み出され、動き補償回路８７で可変長復号回路８２より出力された動きベクトルに対応する動き補償が施される。そして、演算器８５において、IDCT回路８４より供給された画像データ（差分のデータ）と加算され、出力される。この加算されたデータ、すなわち、復号されたPピクチャのデータは、演算器８５に後に入力される画像データ（BピクチャまたはPピクチャのデータ）の予測画像データ生成のために、フレームメモリ８６の後方予測画像部８６ｂに供給されて記憶される。

Pピクチャのデータであっても、画像内予測モードのデータは、Iピクチャのデータと同様に、演算器８５において処理は行われず、そのまま後方予測画像部８６ｂに記憶される。

このPピクチャは、次のBピクチャの次に表示されるべき画像であるため、この時点では、まだフォーマット変換回路３２へ出力されない（上述したように、Bピクチャの後に入力されたPピクチャが、Bピクチャより先に処理され、伝送されている）。

IDCT回路８４より供給された画像データが、Bピクチャのデータである場合、可変長復号回路８２より供給された予測モードに対応して、フレームメモリ８６の前方予測画像部８６ａに記憶されているIピクチャの画像データ（前方予測モードの場合）、後方予測画像部８６ｂに記憶されているPピクチャの画像データ（後方予測モードの場合）、または、その両方の画像データ（両方向予測モードの場合）が読み出され、動き補償回路８７において、可変長復号回路８２より出力された動きベクトルに対応する動き補償が施されて、予測画像が生成される。但し、動き補償を必要としない場合（画像内予測モードの場合）、予測画像は生成されない。

このようにして、動き補償回路８７で動き補償が施されたデータは、演算器８５において、IDCT回路８４の出力と加算される。この加算出力は、フォーマット変換回路３２に出力される。

ただし、この加算出力はBピクチャのデータであり、他の画像の予測画像生成

のために利用されることがないため、フレームメモリ 86 には記憶されない。

Bピクチャの画像が出力された後、後方予測画像部 86b に記憶されている Pピクチャの画像データが読み出され、動き補償回路 87 を介して演算器 85 に供給される。但し、このとき、動き補償は行われず。

なお、このデコーダ 31 には、図 5 のエンコーダ 18 における予測モード切り替え回路 52 と DCT モード切り替え回路 55 に対応する回路が図示されていないが、これらの回路に対応する処理、すなわち、奇数フィールドと偶数フィールドのラインの信号が分離された構成を元の構成に必要なに応じて戻す処理は、動き補償回路 87 により実行される。

また、上述した説明においては、輝度信号の処理について説明したが、色差信号の処理も同様に行われる。ただし、この場合の動きベクトルは、輝度信号用の動きベクトルを、垂直方向および水平方向に $1/2$ にしたものが用いられる。

図 13 は、符号化された画像の品質を示している。画像の品質 (SNR: Signal to Noise Ratio) は、ピクチャタイプに対応して制御され、Iピクチャ、および Pピクチャは高品質とされ、Bピクチャは、I、Pピクチャに比べて劣る品質とされて伝送される。これは、人間の視覚特性を利用した手法であり、全ての画像品質を平均化するよりも、品質を振動させたほうが視覚上の画質が良くなるためである。このピクチャタイプに対応した画質の制御は、図 7 の量子化回路 57 により実行される。

図 14 及び図 15 は、本発明を適用したトランスコーダ 101 の構成を示しており、図 15 は、図 14 のさらに詳細な構成を示している。このトランスコーダ 101 は、復号装置 102 に入力された符号化ビデオビットストリーム

(encoded video bit stream) の GOP 構造及びビットレートを、オペレータの所望する GOP 構造及びビットレートに変換する。このトランスコーダ 101 の機能を説明するために、図 15 には図示されていないが、このトランスコーダ 101 の前段に、このトランスコーダ 101 とほぼ同様の機能を有した 3 つのトランスコーダが接続されているものとする。つまり、ビットストリームの GOP 構造及び

ビットレートをさまざまに変更するために、第1のトランスコード、第2のトランスコード、および第3のトランスコードが順に直列に接続され、その第3のトランスコードの後ろに、この図15に示された第4のトランスコードが接続されているものとする。

本発明の以下の説明において、この第1のトランスコードにおいて行われた符号化処理を第1世代の符号化処理と定義し、第1のトランスコードの後ろに接続された第2のトランスコードにおいて行われた符号化処理を第2世代の符号化処理と定義し、第2のトランスコードの後ろに接続された第3のトランスコードにおいて行われた符号化処理を第3世代の符号化処理と定義し、第3のトランスコードの後ろに接続された第4のトランスコード（図15に示されたトランスコード101）において行われる符号化処理を第4世代の符号化処理または現在の符号化処理と定義することにする。

また、第1世代の符号化処理において生成された符号化パラメータを第1世代の符号化パラメータと呼び、第2世代の符号化処理において生成された符号化パラメータを第2世代の符号化パラメータと呼び、第3世代の符号化処理において生成された符号化パラメータを第3世代の符号化パラメータと呼び、第4世代の符号化処理において生成された符号化パラメータを第4世代の符号化パラメータまたは現在の符号化パラメータと呼ぶことにする。

まず、この図15に示されたトランスコード101に供給される符号化ビデオストリームST (3rd) について説明する。ST (3rd) は、このトランスコード101の前段に設けられている第3のトランスコードにおける第3世代の符号化処理において生成された第3世代の符号化ストリームであることを表わしている。この第3世代の符号化処理において生成された符号化ビデオストリームST (3rd) には、第3の符号化処理において生成された第3世代の符号化パラメータが、この符号化ビデオストリームST (3rd) のシーケンス層、GOP層、ピクチャ層、スライス層、及びマクロブロック層に、sequence_header() 関数、sequence_extension() 関数、group_of_pictures_header() 関数、

picture_header()関数、picture_coding_extension()関数、picture_data()関数、slice()関数、及びmacroblock()関数として記述されている。このように第3の符号化処理によって生成された第3の符号化ストリームに、第3の符号化処理において使用した第3の符号化パラメータを記述することはMPEG2規格において定義されていることであって、何ら新規性は無い。

本発明のトランスコーダ101におけるユニークな点は、この第3の符号化ストリームST (3rd) 中に、第3の符号化パラメータを記述するだけでなく、第1世代及び第2世代の符号化処理において生成された第1世代及び第2世代の符号化パラメータも記述されているという点である。

具体的には、この第1世代及び第2世代の符号化パラメータは、第3世代の符号化ビデオストリームST (3rd) のピクチャ層のユーザデータエリアに、ヒストリストリームhistory_stream()として記述されている。本発明においては、第3世代の符号化ビデオストリームST (3rd) のピクチャ層のユーザデータエリアに記述されているヒストリストリームを、「ヒストリ情報」、または「履歴情報」と呼び、このヒストリストリームとして記述されている符号化パラメータを「ヒストリパラメータ」、または「履歴パラメータ」と呼んでいる。

また別の呼び方として、第3世代の符号化ストリームST (3rd) に記述されている第3世代の符号化パラメータを「現在の符号化パラメータ」と呼んだ場合には、第3世代の符号化処理からみて第1世代及び第2世代の符号化処理は、過去に行なわれた符号化処理であるので、第3世代の符号化ストリームST

(3rd) のピクチャ層のユーザデータエリアに記述されているヒストリストリームとして記述されている符号化パラメータを「過去の符号化パラメータ」とも呼んでいる。

このように、この第3の符号化ストリームST (3rd) 中に、第3の符号化パラメータを記述するだけでなく、第1世代及び第2世代の符号化処理において生成された第1世代及び第2世代の符号化パラメータを記述する理由は、トランスコーディング処理によって符号化ストリームのGOP構造やビットレートの変更を

繰り返したとしても、画質劣化を防止することができるからである。

例えば、あるピクチャを第1世代の符号化処理においてPピクチャとして符号化し、第1世代の符号化ストリームのGOP構造を変更するために、第2世代の符号化処理においてそのピクチャをBピクチャとして符号化し、第2世代の符号化ストリームのGOP構造をさらに変更するために、第3世代の符号化処理において、再度そのピクチャをPピクチャとして符号化することが考えられる。MPEG規格に基づく符号化処理及び復号処理は100%可逆の処理ではないので、符号化及び復号処理を繰り返す毎に画質が劣化していくことは知られている。

このような場合に、第3の世代の符号化処理において、量子化スケール、動きベクトル、予測モードなどの符号化パラメータをもう一度計算するのではなくて、第1世代の符号化処理において生成された量子化スケール、動きベクトル、予測モードなどの符号化パラメータを再利用する。第3世代の符号化処理によって新しく生成された量子化スケール、動きベクトル、予測モードなどの符号化パラメータよりも、第1世代の符号化処理によって新しく生成された量子化スケール、動きベクトル、予測モードなどの符号化パラメータの方が、明らかに精度が良いので、この第1世代のパラメータを再利用することによって、符号化及び復号処理を繰り返したとしても画質劣化を少なくすることができる。

上述した本発明にかかる処理を説明するために、図15に示された第4世代のトランスコード101の処理を例に挙げてより詳しく説明する。

復号装置102は、第3世代の符号化ビットストリームST (3rd) に含まれている符号化ビデオを第3世代の符号化パラメータを使用して復号し、復号されたベースバンドのデジタルビデオデータを生成するための装置である。さらに、復号装置102は、第3世代の符号化ビットストリームST (3rd) のピクチャ層のユーザデータエリアにヒストリストリームとして記述されている第1世代及び第2世代の符号化パラメータをデコードするための装置でもある。

具体的には、図16に示されているように、復号装置102は、図5の復号装置2のデコーダ31 (図12) と基本的に同様の構成とされ、供給されたビット

ストリームをバッファリングするための受信バッファ 8 1、符号化ビットストリームを可変長復号するための可変長復号回路 1 1 2、可変長復号されたデータを可変長復号回路 1 1 2 から供給された量子化スケールに従って逆量子化する逆量子化回路 8 3、逆量子化された D C T 係数を逆離散コサイン変換する IDCT 回路 8 4、及び動き補償処理を行うための演算器 8 5、フレームメモリ 8 6 及び動き補償回路 8 7 を備えている。

可変長復号回路 1 1 2 は、第 3 世代の符号化ビットストリーム S T (3rd) を復号処理するために、この第 3 世代の符号化ビットストリーム S T (3rd) のピクチャ層、スライス層及びマクロブロック層に記述されている第 3 世代の符号化パラメータを抽出する。たとえば、この可変長復号回路 1 1 2 において抽出される第 3 世代の符号化パラメータは、ピクチャタイプを示す `picture_coding_type`、量子化スケールステップサイズを示す `quantiser_scale_code`、予測モードを示す `macroblock_type`、動きベクトルを示す `motion_vector`、Frame 予測モードか Field 予測モードかを示す `frame/field_motion_type`、及び Frame D C T モードか Field D C T モードかを示す `dct_type` 等である。この可変長復号回路 1 1 2 において抽出された `quantiser_scale_code` は、逆量子化回路 8 3 に供給され、`picture_coding_type`、`quantiser_scale_code`、`macroblock_type`、`motion_vector`、`frame/field_motion_type`、`dct_type` 等のパラメータは、動き補償回路 8 7 に供給される。

可変長復号回路 1 1 2 は、第 3 世代の符号化ビットストリーム S T (3rd) を復号処理するために必要なこれらの符号化パラメータだけではなく、後段の第 5 世代のトランスコードに第 3 世代のヒストリ情報として伝送されるべき符号化パラメータを、第 3 世代の符号化ビットストリーム S T (3rd) のシーケンス層、GOP 層、ピクチャ層、スライス層、及びマクロブロック層から抽出する。もちろん、第 3 世代の復号処理に使用された `picture_coding_type`、`quantiser_scale_code`、`macroblock_type`、`motion_vector`、`frame/field_motion_type`、`dct_type` 等の第 3 世代の符号化パラメータは、この

第3世代の履歴情報に含まれている。履歴情報としてどのような符号化パラメータを抽出するかについては、伝送容量などに応じてオペレータやホストコンピュータ側からあらかじめ設定されている。

さらに、可変長復号回路112は、第3世代の符号化ビットストリームST (3rd) のピクチャ層のユーザデータエリアに記述されているユーザデータを抽出し、そのユーザデータを履歴デコーディング装置104に供給する。

この履歴デコーディング装置104は、第3世代の符号化ビットストリームST (3rd) のピクチャ層に記述されていたユーザデータから、履歴情報として記述されている第1世代の符号化パラメータ及び第2世代の符号化パラメータ（直前の世代よりさらに前の世代の符号化パラメータ）を抽出するための回路である。具体的には、履歴デコーディング装置104は、受け取ったユーザデータのシンタックスを解析することによって、ユーザデータの中に記述されている固有のHistory_Data_Idを検出し、これによって、converted_history_stream()を抽出することができる。さらに、履歴デコーディング装置104は、converted_history_stream()中にある所定間隔に挿入されている1ビットのマーカービット(marker_bit)を取りさることによって、history_stream()を得、そして、そのhistory_stream()のシンタックスを解析することによって、history_stream()中に記述されている第1世代及び第2世代の符号化パラメータを得ることができる。この履歴デコーディング装置104の詳しい動作については、後述する。

履歴情報多重化装置103は、第1世代、第2世代及び第3世代の符号化パラメータを、第4世代の符号化処理を行う符号化装置106に供給するために、復号装置102においてデコードされたベースバンドのビデオデータに、これらの第1世代、第2世代及び第3世代の符号化パラメータを多重化するための回路である。具体的には、履歴情報多重化装置103は、復号装置102の演算器85から出力されたベースバンドのビデオデータ、復号装置102の可変長復号装置112から出力された第3世代の符号化パラメータ、並びに、履歴

デコーディング装置 104 から出力された第 1 世代の符号化パラメータと第 2 世代の符号化パラメータとを受け取り、このベースバンドのビデオデータに、これらの第 1 世代、第 2 世代及び第 3 世代の符号化パラメータを多重化する。第 1 世代、第 2 世代及び第 3 世代の符号化パラメータが多重化されたベースバンドのビデオデータは、伝送ケーブルを介してヒストリ情報分離装置 105 に供給される。

次に、これらの第 1 世代、第 2 世代及び第 3 世代の符号化パラメータのベースバンドビデオデータへの多重化の方法について、図 17 及び図 18 を参照して説明する。図 17 は、MPEG 規格において定義されている、16 ピクセル×16 ピクセルからなる 1 つのマクロブロックを示している。この 16 ピクセル×16 ピクセルのマクロブロックは、輝度信号に関しては 4 つの 8 ピクセル×8 ピクセルからなるサブブロック (Y[0], [1], [2] 及び Y[3]) と、色差信号に関しては 4 つの 8 ピクセル×8 ピクセルからなるサブブロック (Cr[0], r[1], b[0], 及び Cb[1]) から構成されている。

図 18 は、ビデオデータのあるフォーマットを表している。このフォーマットは、ITU 勧告-RDT 601 において定義されているフォーマットであって、放送業界において使用されている所謂「D1 フォーマット」をあらわしている。この D1 フォーマットは、10 ビットのビデオデータを伝送するためのフォーマットとして規格化されたので、ビデオデータの 1 ピクセルを 10 ビットで表現できるようになっている。

MPEG 規格によってデコードされたベースバンドのビデオデータは 8 ビットであるので、本発明のトランスコーダにおいては、図 18 に示したように、D1 フォーマットの 10 ビットのうち上位 8 ビット (D9 乃至 D2) を使用して、MPEG 規格にもとづいてデコードされたベースバンドのビデオデータを伝送するようにしている。このように、復号された 8 ビットのビデオデータを D1 フォーマットに書き込むと、下位 2 ビット (D1 と D0) は、空きビット (unallocated bits) となる。本発明のトランスコーダではこの空きエリア (unallocated

area) を利用して、ヒストリ情報を伝送するようにしている。

この図 1 8 に記載されたデータブロックは、各サブブロック (Y[0], Y[1], Y[2], Y[3], Cr[0], Cr[1], Cb[0], Cb[1]) における 1 ピクセルを伝送するためのデータブロックであるので、1 マクロブロックのデータを伝送するためには、この図 1 8 に示されているデータブロックが 64 個伝送される。下位 2 ビット (D1 と D0) を使用すれば、1 マクロブロックのビデオデータに対して、合計で 1024 ($=16 \times 64$) ビットのヒストリ情報を伝送できる。従って、1 世代分のヒストリ情報は、256 ビットとなるように生成されているので、過去の 4 ($=1024 / 256$) 世代分のヒストリ情報を 1 マクロブロックのビデオデータに対して重畳することができる。図 1 8 に示した例では、第 1 世代のヒストリ情報、第 2 世代のヒストリ情報、並びに、第 3 世代のヒストリ情報が重畳されている。

ヒストリ情報分離装置 105 は、D1 フォーマットとして伝送されたデータの上位 8 ビットから、ベースバンドビデオデータを抽出し、下位 2 ビットからヒストリ情報を抽出するための回路である。図 1 5 に示した例では、ヒストリ情報分離装置 105 は、伝送データからベースバンドのビデオデータを抽出して、そのビデオデータを符号化装置 106 に供給するとともに、伝送データから第 1 世代、第 2 世代及び第 3 世代のヒストリ情報を抽出して、符号化装置 106 とヒストリエンコーディング装置 107 にそれぞれ供給する。

符号化装置 106 は、ヒストリ情報分離装置 105 から供給されたベースバンドのビデオデータを、オペレータまたはホストコンピュータから指定された GOP 構造及びビットレートを有するビットストリームになるように符号化するための装置である。なお、GOP 構造を変更するとは、たとえば、GOP に含まれるピクチャの数、I ピクチャと I ピクチャの間に存在する P ピクチャの数、及び I ピクチャと P ピクチャ (または I ピクチャ) の間に存在する B ピクチャの数を変更することを意味する。

図 1 5 に示された例では、供給されたベースバンドのビデオデータには、第 1

世代、第2世代及び第3世代のヒストリ情報が重畳されているので、この符号化装置106は、再符号化処理による画質劣化が少なくなるように、これらのヒストリ情報を選択的に再利用して第4世代の符号化処理を行う。

図19は、この符号化装置106に設けられている符号化装置106の具体的な構成を示している図である。この符号化装置106は、基本的には、図7に示したエンコーダ18と同様に構成され、動きベクトル検出回路50、フレーム／フィールド予測モード切り替え回路52、演算器53、DCTモード切り替え回路55、DCT回路56、量子化回路57、可変長符号化回路58、伝送バッファ59、逆量子化回路60、逆DCT回路61、演算器62、フレームメモリ63、並びに動き補償回路64を備えている。これらの、各回路の機能は、図7において説明したエンコーダ18における場合の機能とほぼ同じであるので、その説明は省略する。以下に、この符号化装置106と、図7において説明したエンコーダ18との異なる点を中心に説明する。

この符号化装置106は、上述した各回路の動作及び機能を制御するためのコントローラ70を有している。このコントローラ70は、オペレータまたはホストコンピュータからGOP構造に関するインストラクションを受け取って、そのGOP構造に対応するように各ピクチャのピクチャタイプを決定する。また、このコントローラ70は、オペレータまたはホストコンピュータからターゲットビットレートの情報を受け取り、この符号化装置106から出力されるビットレートがこの指定されたターゲットビットレートになるように、量子化回路57を制御する。

さらに、このコントローラ70は、ヒストリ情報分離装置105から出力された複数世代のヒストリ情報を受け取り、これらのヒストリ情報を再利用して参照ピクチャの符号化処理を行う。以下に詳しく説明する。

まず、このコントローラ70は、オペレータによって指定されたGOP構造から決定された参照ピクチャのピクチャタイプと、ヒストリ情報に含まれるピクチャタイプが一致するか否かを判断する。つまり、指定されたピクチャタイプと同じ

ピクチャタイプでこの参照ピクチャが過去において符号化されたことがあるか否かを判断する。

図15に示された例をあげてよりわかりやすく説明するのであれば、このコントローラ70は、第4世代の符号化処理としてこの参照ピクチャにアサインされたピクチャタイプが、第1世代の符号化処理におけるこの参照ピクチャのピクチャタイプ、第2世代の符号化処理におけるこの参照ピクチャのピクチャタイプ、または第3世代の符号化処理におけるこの参照ピクチャのピクチャタイプのいずれかと一致するか否かを判断する。

もし、第4世代の符号化処理としてこの参照ピクチャに指定されたピクチャタイプが、過去の符号化処理におけるどのピクチャタイプとも一致しないのであれば、このコントローラ70は、「通常符号化処理」を行う。つまり、この場合には、第1世代、第2世代または第3世代のどの世代の符号化処理においても、第4世代の符号化処理としてアサインされたピクチャタイプで、この参照ピクチャが符号化処理されたことがないということになる。一方、もし、第4世代の符号化処理としてこの参照ピクチャに指定されたピクチャタイプが、過去の符号化処理におけるいずれかのピクチャタイプと一致するのであれば、このコントローラ70は、「パラメータ再利用符号化処理」を行う。つまり、この場合には、第1世代、第2世代または第3世代のいずれかの世代の符号化処理において、第4世代の符号化処理としてアサインされたピクチャタイプで、この参照ピクチャが符号化処理されたことがあるということになる。

まず、最初にコントローラ70の通常符号化処理について説明する。

動きベクトル検出回路50は、フレーム予測モードまたはフィールド予測モードのどちらが選択されるべきかを判断するために、フレーム予測モードにおける予測誤差とフィールド予測モードにおける予測誤差をそれぞれ検出し、その予測誤差の値をコントローラ70に供給する。コントローラ70は、それらの予測誤差の値を比較し、その予測誤差の値が小さい方の予測モードを選択する。予測モード切り替え回路52は、コントローラ70によって選択された予測モードに対応

するように信号処理を行い、それを演算器 53 に供給する。

具体的には、予測モード切り替え回路 52 は、フレーム予測モードが選択された場合には、図 8 を参照して説明したように、輝度信号に関しては、入力された状態のまま演算器 53 に出力するように信号処理を行い、色差信号に関しては、奇数フィールドラインと偶数フィールドラインとが混在するように信号処理する。一方、フィールド予測モードが選択された場合には、図 9 を参照して説明したように、輝度信号に関しては、輝度ブロック Y[1]と Y[2]を奇数フィールドラインで構成し、輝度ブロック Y[3]と Y[4]を偶数フィールドラインで構成するように信号処理し、色差信号に関しては、上 4 ラインを奇数フィールドラインで構成し、下 4 ラインを偶数フィールドラインで構成するように信号処理する。

さらに、動きベクトル検出回路 50 は、画像内予測モード、前方予測モード、後方予測モード、または両方向予測モードのうちのいずれの予測モードを選択するかを決定するために、各予測モードにおける予測誤差を生成し、各予測モードにおける予測誤差をコントローラ 70 にそれぞれ供給する。コントローラ 70 は、前方予測、後方予測および両方向予測の予測誤差のうちの最も小さいものを、インタ予測の予測誤差として選択する。さらに、このインタ予測の予測誤差と、画像内予測の予測誤差とを比較し、その小さい方を選択し、この選択した予測誤差に対応するモードを予測モードとして選択する。すなわち、画像内予測の予測誤差の方が小さければ、画像内予測モードが設定される。インタ予測の予測誤差の方が小さければ、前方予測、後方予測または両方向予測モードのうちの対応する予測誤差が最も小さかったモードが設定される。コントローラ 70 は、選択した予測モードに対応するように、演算器 53 及び動き補償回路 64 を制御する。

DCTモード切り替え回路 55 は、フレーム DCT モードまたはフィールド DCT モードのいずれかを選択するために、4 個の輝度ブロックのデータを、奇数フィールドラインと偶数フィールドラインが混在するような信号形態（フレーム DCT モード）に変換するとともに、奇数フィールドラインと偶数フィールドラインが分離された信号形態（フィールド DCT モード）に変換して、それぞれの信号を D

CT回路56に供給する。DCT回路56は、奇数フィールドと偶数フィールドを混在してDCT処理した場合における符号化効率と、奇数フィールドと偶数フィールドを分離した状態においてDCT処理した場合の符号化効率を計算し、その結果をコントローラ70に供給する。コントローラ70は、DCT回路56から供給されたそれぞれの符号化効率を比較し、符号化効率の良い方のDCTモードを選択し、その選択したDCTモードとなるようにDCTモード切り替え回路55を制御する。

コントローラ70は、オペレータまたはホストコンピュータから供給された目標ビットレートを示すターゲットビットレートと、送信バッファ59にバッファリングされているビット量を示す信号、つまり、バッファ残量を示す信号を受け取り、このターゲットビットレートとバッファ残量に基づいて、量子化回路57の量子化ステップサイズをコントロールするための `feedback_q_scale_code` を生成する。この `feedback_q_scale_code` は、この送信バッファ59がオーバーフローまたはアンダーフローしないように、この送信バッファ59のバッファ残量に応じて生成される制御信号であって、また、送信バッファ59から出力されるビットストリームのビットレートが、ターゲットビットレートになるように制御する信号でもある。

具体的には、例えば、送信バッファ59にバッファリングされているビット量が少なくなってしまった場合には、次に符号化するピクチャの発生ビット量が増えるように、量子化ステップサイズを小さくし、一方、送信バッファ59にバッファリングされているビット量が多くなってしまった場合には、次に符号化するピクチャの発生ビット量が少なくなるように、量子化ステップサイズを大きくする。なお、`feedback_q_scale_code` と量子化ステップサイズは比例し、`feedback_q_scale_code` を大きくすると、量子化ステップサイズは大きくなり、`feedback_q_scale_code` を小さくすると、量子化ステップサイズは小さくなる。

次に、このトランスコーダ101の特徴の1つでもある、パラメータ再利用符号化処理について説明する。この処理をより分かりやすく説明するために、参照

ピクチャは、第1世代の符号化処理においてPピクチャとして符号化され、第2世代の符号化処理においてIピクチャとして符号化処理され、第3世代の符号化処理においてBピクチャとして符号化されていたものとし、今回の第4世代の符号化処理において、この参照ピクチャをPピクチャとして符号化しなければいけないものとする。

この場合には、第4世代のピクチャタイプとしてアサインされたピクチャタイプと同じピクチャタイプ（Iピクチャ）で、この参照ピクチャは第1世代の符号化処理において符号化されているので、コントローラ70は、供給されたビデオデータから符号化パラメータを新しく作成するのではなく、第1世代の符号化パラメータを使用して符号化処理を行う。この第4の符号化処理において再利用する符号化パラメータは、代表的なパラメータとしては、量子化スケールステップサイズを示す `quantiser_scale_code`、予測方向モードを示す `macroblock_type`、動きベクトルを示す `motion_vector`、Frame 予測モードか Field 予測モードかを示す `frame/field_motion_type`、及び Frame DCT モードか Field DCT モードかを示す `dct_type` 等である。

コントローラ70は、ヒストリ情報として伝送されたすべての符号化パラメータを再利用するわけではなく、再利用した方が望ましいと想定される上述したような符号化パラメータについては再利用し、再利用しない方が望ましいと考えられる符号化パラメータについては、新しく生成する。

次に、この符号化パラメータ再利用符号化処理について、上述した通常符号化処理と異なる点を中心に説明する。

動きベクトル検出回路50は、上述した通常符号化処理においては、参照ピクチャの動きベクトルの検出を行ったが、このパラメータ再利用符号化処理においては、動きベクトル `motion_vector` の検出処理は行わずに、第1世代のヒストリ情報として供給された動きベクトル `motion_vector` を再利用する。その理由について説明する。

第3世代の符号化ストリームを復号したベースバンドのビデオデータは、少な

くとも3回の復号及び符号化処理が行われているので、オリジナルビデオデータに比べると、明らかに画質が劣化している。画質が劣化しているビデオデータから動きベクトルを検出したとしても、正確な動きベクトルは検出できない。つまり、第4世代の符号化処理において検出された動きベクトルよりも、第1世代のヒストリ情報として供給されている動きベクトルの方が、明らかに、精度の高い動きベクトルである。つまり、第1世代の符号化パラメータとして伝送された動きベクトルを再利用することによって、第4世代の符号化処理を行ったとしても画質が劣化しない。コントローラ70は、この第1世代のヒストリ情報として供給された動きベクトル `motion_vector` を、第4世代の符号化処理において符号化されるこの参照ピクチャの動きベクトル情報として、動き補償回路64及び可変長符号化回路58に供給する。

さらに、動きベクトル検出回路50は、フレーム予測モードとフィールド予測モードのどちらが選択されるかを判断するために、フレーム予測モードにおける予測誤差とフィールド予測モードにおける予測誤差をそれぞれ検出したが、このパラメータ再利用符号化処理においては、このフレーム予測モードにおける予測誤差とフィールド予測モードにおける予測誤差を検出する処理は行わずに、第1世代のヒストリ情報として供給されている `Frame` 予測モードか `Field` 予測モードかを示す `frame/field_motion_type` を再利用する。なぜなら、第4世代の符号化処理において検出された各予測モードにおける予測誤差よりも、第1世代において検出された各予測モードにおける予測誤差の方が精度が高いため、精度の高い予測誤差によって決定された予測モードを選択した方がより最適な符号化処理を行うことができるからである。

具体的には、コントローラ70は、この第1世代のヒストリ情報として供給されている `frame/field_motion_type` に対応する制御信号を予測モード切り替え回路52に供給し、予測モード切り替え回路52は、この再利用された `frame/field_motion_type` に対応した信号処理を行う。

さらには、動きベクトル検出回路50は、通常符号化処理においては、画像内

予測モード、前方予測モード、後方予測モード、または両方向予測モードのうちのいずれの予測モード（以下、この予測モードを、予測方向モードとも称する）を選択するかを決定するために、各予測方向モードにおける予測誤差を計算していたが、このパラメータ再利用符号化処理においては、各予測方向モードにおける予測誤差の計算は行わず、第1世代のヒストリ情報として供給された `macroblock_type` に基づいて予測方向モードを決定する。なぜなら、第4世代の符号化処理における各予測方向モードにおける予測誤差よりも、第1世代の符号化処理における各予測方向モードにおける予測誤差の方がより精度が高いため、より精度の高い予測誤差によって決定された予測方向モードを選択した方が、より効率の良い符号化処理が行えるからである。具体的には、コントローラ70は、第1世代のヒストリ情報に含まれている `macroblock_type` によって示される予測方向モードを選択し、その選択した予測方向モードに対応するように、演算器53及び動き補償回路64をコントロールする。

DCTモード切り替え回路55は、通常符号化処理においては、フレームDCTモードの符号化効率と、フィールドDCTモードの符号化効率を比較するために、フレームDCTモードの信号形態に変換した信号と、フィールドDCTモードの信号形態に変換した信号の両方をDCT回路56に供給していたが、このパラメータ再利用符号化処理では、フレームDCTモードの信号形態に変換した信号と、フィールドDCTモードの信号形態に変換した信号の両方を生成する処理は行っておらず、第1世代のヒストリ情報に含まれている `dct_type` によって示されたDCTモードに対応する処理のみを行っている。具体的には、コントローラ70は、第1世代のヒストリ情報に含まれている `dct_type` を再利用し、DCTモード切り替え回路55がこの `dct_type` によって示されるDCTモードに対応した信号処理を行うように、DCTモード切り替え回路55をコントロールする。

コントローラ70は、通常符号化処理では、オペレータによって指定されたターゲットビットレートと送信バッファ残量に基づいて、量子化回路57の量子化ステップサイズをコントロールしていたが、このパラメータ再利用符号化処理で

は、ターゲットビットレート、送信バッファ残量及びヒストリ情報に含まれている過去の量子化スケールに基づいて、量子化回路 57 の量子化ステップサイズをコントロールする。なお、以下の説明において、ヒストリ情報に含まれている過去の量子化スケールを `history_q_scale_code` と記述することにする。また、後述するヒストリストリームにおいては、この量子化スケールを、`quantiser_scale_code` と記述している。

まず、コントローラ 70 は、通常符号化処理と同じように、現在の量子化スケール `feedback_q_scale_code` を生成する。この `feedback_q_scale_code` は、この送信バッファ 59 がオーバーフロー及びアンダーフローしないように、この送信バッファ 59 のバッファ残量に応じて決定される値である。続いて、第 1 世代のヒストリストリームに含まれている過去の量子化スケール `history_q_scale_code` の値と、この現在の量子化スケール `feedback_q_scale_code` の値を比較し、どちらの量子化スケールの方が大きいかを判断する。量子化スケールが大きいとは、量子化ステップが大きいことを意味する。もし、現在の量子化スケール `feedback_q_scale_code` が、過去の量子化スケール `history_q_scale_code` よりも大きいのであれば、コントローラ 70 は、この現在の量子化スケール `feedback_q_scale_code` を量子化回路 57 に供給する。一方、過去の量子化スケール `history_q_scale_code` が、現在の量子化スケール `feedback_q_scale_code` よりも大きいのであれば、コントローラ 70 は、この過去の量子化スケール `history_q_scale_code` を量子化回路 57 に供給する。

つまり、コントローラ 70 は、ヒストリ情報に含まれている複数の過去の量子化スケールと、送信バッファの残量から計算された現在の量子化スケールの中で、もっとも大きい量子化スケールコードを選択する。また、別の言葉で説明するのであれば、コントローラ 70 は、過去（第 1、第 2 及び第 3 世代）の符号化処理における量子化ステップまたは現在（第 4 世代）の符号化処理において使用された量子化ステップの中で、もっとも大きい量子化ステップを使用して量子化を行うように量子化回路 57 を制御する。この理由を以下に説明する。

たとえば、第3世代の符号化処理において生成されたストリームのビットレートが4 [Mbps]であって、この第4世代の符号化処理を行う符号化装置106に対して設定されたターゲットビットレートが15 [Mbps]であったとする。このときに、ターゲットビットレートが上がっているので、単純に量子化ステップを小さくすれば良いかという、実際にはそうではない。過去の符号化処理において大きい量子化ステップで符号化処理されたピクチャを、現在の符号化処理において、量子化ステップを小さくして符号化処理を行ったとしても、このピクチャの画質は向上することはない。つまり、過去の符号化処理における量子化ステップよりも小さい量子化ステップで符号化することは、単にビット量が増えるだけであって、画質を向上させることにはならない。よって、過去（第1、第2及び第3世代）の符号化処理における量子化ステップまたは現在（第4世代）の符号化処理において使用された量子化ステップの中で、もっとも大きい量子化ステップを使用して量子化を行うと、もっとも効率の良い符号化処理が行える。

次に、図15におけるヒストリデコーディング装置104とヒストリエンコーディング装置107についてさらに説明する。尚、図15においては、このヒストリデコーディング装置104が、復号化装置102とは、別の回路又は装置のように表現されているが、これはヒストリデコーディング装置104の機能及び構成をよりわかりやすく説明するために、復号化装置102とは別のブロックのように表現しただけであって、実際には、ヒストリデコーディング装置104の処理は、復号化装置102の可変長復号化回路及び復号化制御回路（復号化コントローラ）の中で行われている処理である。同様に、図15においては、このヒストリエンコーディング装置107が、符号化装置106とは、別の回路又は装置のように表現されているが、これはヒストリエンコーディング装置107の機能及び構成をよりわかりやすく説明するために、符号化装置106とは別のブロックのように表現しただけであって、実際には、ヒストリエンコーディング装置107の処理は、符号化装置102の可変長符号化回路及び符号化制御回路（符号化コントローラ）の中で行われている処理である。

図15に示すように、ヒストリデコーディング装置104は、復号装置102より供給されるユーザデータをデコードするユーザデータデコーダ201、ユーザデータデコーダ201の出力を変換するコンバータ202、およびコンバータ202の出力から履歴情報を再生するヒストリ VLD203により構成されている。

また、ヒストリエンコーディング装置107は、ヒストリ情報分離装置105より供給される3世代分の符号化パラメータをフォーマット化するヒストリ VLC211、ヒストリ VLC211の出力を変換するコンバータ212、コンバータ212の出力をユーザデータのフォーマットにフォーマットするユーザデータフォーマッタ213により構成されている。

ユーザデータデコーダ201は、復号装置102より供給されるユーザデータをデコードして、コンバータ202に出力する。詳細は図51を参照して後述するが、ユーザデータ (user_data()) は、user_data_start_code と user_data からなり、MPEG 規格においては user_data の中に、連続する23ビットの”0”

(start_code と同一のコード) を発生させることを禁止している。これは、そのデータが、start_code として誤検出されるのを防止するためである。履歴情報 (history_stream()) は、ユーザデータエリアに (MPEG 規格の user_data の一種として) 記述され、その中には、このような連続する23ビット以上の”0”が存在することがあり得るので、これを、連続する23ビット以上の”0”が発生しないように、所定のタイミングで”1”を挿入処理して、converted_history_stream() (後述する図38) に変換する必要がある。この変換を行うのは、ヒストリエンコーディング装置107のコンバータ212である。ヒストリデコーディング装置104のコンバータ202は、このコンバータ212と逆の変換処理を行う (連続する23ビット以上の”0”を発生させないために挿入された”1”を除去する) ものである。

ヒストリ VLD203は、コンバータ202の出力から履歴情報 (いまの場合、第1世代の符号化パラメータと第2世代の符号化パラメータ) を生成し、ヒスト

り情報多重化装置 103 に出力する。

一方、ヒストリエンコーディング装置 107 においては、ヒストリ VLC 211 がヒストリ情報分離装置 105 より供給される 3 世代分の（第 1 世代、第 2 世代、および第 3 世代の）符号化パラメータを履歴情報のフォーマットに変換する。このフォーマットには、固定長のもの（後述する図 40 乃至図 46）と、可変長のもの（後述する図 47 以降）とがある。これらの詳細については後述する。

ヒストリ VLC 211 により、フォーマット化された履歴情報は、コンバータ 212 において、converted_history_stream() に変換される。これは、上述したように、user_data() の start_code が誤検出されないようにするための処理である。すなわち、履歴情報内には連続する 23 ビット以上の "0" が存在するが、user_data 中には連続する 23 ビット以上の "0" を配置することができないので、この禁止項目に触れないようにコンバータ 212 によりデータを変換する（"1" を所定のタイミングで挿入する）のである。

ユーザデータフォーマッタ 213 は、コンバータ 212 より供給される converted_history_stream() に、後述する図 38 に基づいて、History_Data_ID を付加し、さらに、user_data_stream_code を付加して、video stream 中に挿入できる MPEG 規格の user_data を生成し、符号化装置 106 に出力する。

図 20 は、ヒストリ VLC 211 の構成例を表している。その符号語変換器 301 と符号長変換器 305 には、符号化パラメータ（今回、履歴情報として伝送する符号化パラメータ）（項目データ）と、この符号化パラメータを配置するストリームを特定する情報（例えば、シンタックスの名称（例えば、後述する sequence_header の名称））（項目 NO.）が、ヒストリ情報分離装置 105 から供給されている。符号語変換器 301 は、入力された符号化パラメータを、指示されたシンタックスに対応する符号語に変換し、バレルシフタ 302 に出力する。バレルシフタ 302 は、符号語変換器 301 より入力された符号語を、アドレス発生回路 306 より供給されるシフト量に対応する分だけシフトし、バイト単位の符号語として、スイッチ 303 に出力する。アドレス発生回路 306 が出力

するビットセレクト信号により切り換えられるスイッチ303は、ビット分設けられており、パレルシフタ302より供給される符号語を、RAM304に供給し、記憶させる。このときの書き込みアドレスは、アドレス発生回路306から指定される。また、アドレス発生回路306から読み出しアドレスが指定されたとき、RAM304に記憶されているデータ（符号語）が読み出され、後段のコンバータ212に供給されるとともに、必要に応じて、スイッチ303を介してRAM304に再び供給され、記憶される。

符号長変換器305は、入力されるシンタックスと符号化パラメータとから、その符号化パラメータの符号長を決定し、アドレス発生回路306に出力する。アドレス発生回路306は、入力された符号長に対応して、上述したシフト量、ビットセレクト信号、書き込みアドレス、または読み出しアドレスを生成し、それらを、それぞれパレルシフタ302、スイッチ303、またはRAM304に供給する。

以上のように、ヒストリVLC211は、いわゆる可変長符号化器として構成され、入力された符号化パラメータを可変長符号化して出力する。

図21は、以上のようにしてヒストリフォーマット化されたデータをデコードするヒストリVLD203の構成例を表している。このヒストリVLD203には、コンバータ202から供給された符号化パラメータのデータがRAM311に供給されて、記憶される。このときの書き込みアドレスは、アドレス発生回路315から供給される。アドレス発生回路315はまた、所定のタイミングで読み出しアドレスを発生し、RAM311に供給する。このとき、RAM311は、読み出しアドレスに記憶されているデータを読み出し、パレルシフタ312に出力する。パレルシフタ312は、アドレス発生回路315が出力するシフト量に対応する分だけ、入力されるデータをシフトし、逆符号長変換器313と逆符号語変換器314に出力する。

逆符号長変換器313にはまた、コンバータ202から、符号化パラメータが配置されているストリームのシンタックスの名称（項目NO.）が供給されている

。逆符号長変換器 3 1 3 は、そのシンタックスに基づいて、入力されたデータ（符号語）から符号長を求め、求めた符号長をアドレス発生回路 3 1 5 に出力する。

また、逆符号語変換器 3 1 4 は、バレルシフタ 3 1 2 より供給されたデータを、シンタックスに基づいて復号し（逆符号語化し）、ヒストリ情報多重化装置 1 0 3 に出力する。

また、逆符号語変換器 3 1 4 は、どのような符号語が含まれているのかを特定するのに必要な情報（符号語の区切りを決定するのに必要な情報）を抽出し、アドレス発生回路 3 1 5 に出力する。アドレス発生回路 3 1 5 は、この情報と逆符号長変換器 3 1 3 より入力された符号長に基づいて、書き込みアドレスおよび読み出しアドレスを発生し、RAM 3 1 1 に出力するとともに、シフト量を発生し、バレルシフタ 3 1 2 に出力する。

図 2 2 は、コンバータ 2 1 2 の構成例を表している。この例においては、ヒストリ VLC 2 1 1 とコンバータ 2 1 2 の間に配置されているバッファメモリ 3 2 0 の、コントローラ 3 2 6 が出力する読み出しアドレスから 8 ビットのデータが読み出され、D 型フリップフロップ（D-FF）3 2 1 に供給され、保持されるようになされている。そして、D 型フリップフロップ 3 2 1 より読み出されたデータは、スタッフ回路 3 2 3 に供給されるとともに、8 ビットの D 型フリップフロップ 3 2 2 にも供給され、保持される。D 型フリップフロップ 3 2 2 より読み出された 8 ビットのデータは、D 型フリップフロップ 3 2 1 より読み出された 8 ビットのデータと合成され、1 6 ビットの平行データとして、スタッフ回路 3 2 3 に供給される。

スタッフ回路 3 2 3 は、コントローラ 3 2 6 より供給されるスタッフ位置を示す信号（stuff position）の位置に符号” 1 ”を挿入し（スタッフイングし）、合計 1 7 ビットのデータとして、バレルシフタ 3 2 4 に出力する。

バレルシフタ 3 2 4 は、コントローラ 3 2 6 より供給されるシフト量を示す信号（shift）に基づいて入力されたデータをシフトして、8 ビットのデータを抽

出し、8ビットのD型フリップフロップ325に出力する。D型フリップフロップ325に保持されたデータは、そこから読み出され、バッファメモリ327を介して、後段のユーザデータフォーマッタ213に供給される。この時、コントローラ326は、出力するデータとともに、書き込みアドレスを発生し、コンバータ212とユーザデータフォーマッタ213との間に介在するバッファメモリ327に供給する。

図23は、スタンプ回路323の構成例を表している。D型フリップフロップ322、321より入力された16ビットのデータは、それぞれスイッチ331-16乃至331-1の接点aに入力されている。スイッチ331-i ($i=0$ 乃至15)の接点cには、MSB側(図中上方)に隣接するスイッチのデータが供給されている。例えば、スイッチ331-12の接点cには、MSB側に隣接するスイッチ331-13の接点aに供給されているLSBから13番目のデータが供給されており、スイッチ331-13の接点cには、MSB側に隣接するスイッチ331-14の接点aに供給されているLSB側から14番目のデータが供給されている。

但し、LSBに対応するスイッチ331-1よりさらに下側のスイッチ331-0の接点aは、開放されている。また、MSBに対応するスイッチ331-16の接点cは、それより上位のスイッチが存在しないため、開放されている。

各スイッチ331-0乃至331-16の接点bには、データ"1"が供給されている。

デコーダ332は、コントローラ326より供給されるデータ"1"を挿入する位置を示す信号 *stuff position* に対応して、スイッチ331-0乃至331-16のうち、1つのスイッチを接点b側に切り替え、それよりLSB側のスイッチは、接点c側にそれぞれ切り替えさせ、それよりMSB側のスイッチは、接点a側に切り替えさせる。

図23は、LSB側から13番目にデータ"1"を挿入する場合の例を示している。従って、この場合、スイッチ331-0乃至スイッチ331-12は、いず

れも接点 c 側に切り替えられ、スイッチ 3 3 1-1 3 は、接点 b 側に切り替えられ、スイッチ 3 3 1-1 4 乃至スイッチ 3 3 1-1 6 は、接点 a 側に切り替えられている。

図 2 2 のコンバータ 2 1 2 は、以上のような構成により、2 2 ビットの符号を 2 3 ビットに変換して、出力することになる。

図 2 4 は、図 2 2 のコンバータ 2 1 2 の各部の出力データのタイミングを表している。コンバータ 2 1 2 のコントローラ 3 2 6 がバイト単位のクロックに同期して、読み出しアドレス (図 2 4 (A)) を発生すると、バッファメモリ 3 2 0 から、それに対応するデータが、バイト単位で読み出され、D 型フリップフロップ 3 2 1 に一旦保持される。そして、D 型フリップフロップ 3 2 1 より読み出されたデータ (図 2 4 (B)) は、スタッフ回路 3 2 3 に供給されるとともに、D 型フリップフロップ 3 2 2 に供給され、保持される。D 型フリップフロップ 3 2 2 に保持されたデータは、そこからさらに読み出され (図 2 4 (C))、スタッフ回路 3 2 3 に供給される。

従って、スタッフ回路 3 2 3 の入力 (図 2 4 (D)) は、読み出しアドレス A 1 のタイミングにおいて、最初の 1 バイトのデータ D 0 とされ、次の読み出しアドレス A 2 のタイミングにおいて、1 バイトのデータ D 0 と 1 バイトのデータ D 1 より構成される 2 バイトのデータとなり、さらに読み出しアドレス A 3 のタイミングにおいては、データ D 1 とデータ D 2 より構成される 2 バイトのデータとなる。

スタッフ回路 3 2 3 には、データ" 1 " を挿入する位置を示す信号 stuff position (図 2 4 (E)) がコントローラ 3 2 6 より供給される。スタッフ回路 3 2 3 のデコーダ 3 3 2 は、スイッチ 3 3 1-1 6 乃至 3 3 1-0 のうち、この信号 stuff position に対応するスイッチを接点 b に切り換え、それより LSB 側のスイッチを接点 c 側に切り換え、さらにそれより MSB 側のスイッチを接点 a 側に切り換える。これにより、データ" 1 " が挿入されるので、スタッフ回路 3 2 3 からは、信号 stuff position で示す位置に、データ" 1 " が挿入されたデー

タ（図 2 4（F））が出力される。

バレルシフト 3 2 4 は、入力されたデータを、コントローラ 3 2 6 より供給される信号 shift（図 2 4（G））で示される量だけバレルシフトして、出力する（図 2 4（H））。この出力がさらに D 型フリップフロップ 3 2 5 で一旦保持された後、後段に出力される（図 2 4（I））。

D 型フリップフロップ 3 2 5 より出力されるデータには、2 2 ビットのデータの次に、データ” 1 ”が挿入されている。従って、データ” 1 ”と、次のデータ” 1 ”の間には、その間のビットが全て 0 であったとしても、0 のデータの連続する数は 2 2 となる。

図 2 5 は、コンバータ 2 0 2 の構成例を表している。このコンバータ 2 0 2 の D 型フリップフロップ 3 4 1 乃至コントローラ 3 4 6 よりなる構成は、図 2 2 に示したコンバータ 2 1 2 の D 型フリップフロップ 3 2 1 乃至コントローラ 3 2 6 と基本的に同様の構成であるが、コンバータ 2 1 2 におけるスタップ回路 3 2 3 に代えて、ディリット回路 3 4 3 が挿入されている点がコンバータ 2 1 2 における場合と異なっている。その他の構成は、図 2 2 のコンバータ 2 1 2 における場合と同様である。

すなわち、このコンバータ 2 0 2 においては、コントローラ 3 4 6 が出力する削除するビットの位置を示す信号 delete position に従って、ディリット回路 3 4 3 が、そのビット（図 2 2 のスタップ回路 3 2 3 で挿入されたデータ” 1 ”）が削除される。

その他の動作は、図 2 2 のコンバータ 2 1 2 における場合と同様である。

図 2 6 は、ディリット回路 3 4 3 の構成例を表している。この構成例においては、D 型フリップフロップ 3 4 2、3 4 1 より入力された 1 6 ビットのデータのうち、LSB 側の 1 5 ビットが、それぞれ対応するスイッチ 3 5 1 - 0 乃至 3 5 1 - 1 4 の接点 a に供給されている。各スイッチの接点 b には、1 ビットだけ MSB 側のデータが供給されている。デコーダ 3 5 2 は、コントローラ 3 4 6 より供給される信号 delete position により指定されるビットを削除して、1 5 ビットの

データとして出力するようになされている。

図 2 6 は、LSB から第 1 3 番目のビットがディリートされる状態を示している。従って、この場合、スイッチ 3 5 1-0 乃至スイッチ 3 5 1-1 1 が接点 a 側に切り替えられ、LSB から第 1 2 番目までの 1 2 ビットが、そのまま選択、出力されている。また、スイッチ 3 5 1-1 2 乃至 3 5 1-1 4 は、それぞれ接点 b 側に切り替えられているので、第 1 4 番目乃至第 1 6 番目のデータが、第 1 3 番目乃至第 1 5 番目のビットのデータとして選択、出力される。

図 2 3 のスタッフ回路 3 2 3 および図 2 6 のディリート回路 3 4 3 の入力 が 1 6 ビットとなっているのは、それぞれ図 2 2 のコンバータ 2 1 2 のスタッフ回路 3 2 3 の入力 が、D 型フリップフロップ 3 2 2, 3 2 1 より供給される 1 6 ビットとされており、また、図 2 5 のコンバータ 2 0 2 においても、ディリート回路 3 4 3 の入力 が、D 型フリップフロップ 3 4 2, 3 4 1 により 1 6 ビットとされているためである。図 2 2 において、スタッフ回路 3 2 3 の出力する 1 7 ビットをバレルシフタ 3 2 4 でバレルシフトすることにより、例えば 8 ビットを最終的に選択、出力しているのと同様に、図 2 5 のコンバータ 2 0 2 においても、ディリート回路 3 4 3 の出力する 1 5 ビットのデータを、バレルシフタ 3 4 4 で所定量だけバレルシフトすることにより、8 ビットのデータとしている。

図 2 7 は、コンバータ 2 1 2 の他の構成例を表している。この構成例においては、カウンタ 3 6 1 が入力データのうち、連続する 0 のビットの数をカウントし、そのカウント結果をコントローラ 3 2 6 に出力するようになされている。コントローラ 3 2 6 は、例えばカウンタ 3 6 1 が連続する 0 のビットを 2 2 個カウントしたとき、信号 *stuff position* をスタッフ回路 3 2 3 に出力する。また、このとき、コントローラ 3 2 6 は、カウンタ 3 6 1 をリセットし、再び連続する 0 のビットの数をカウンタ 3 6 1 にカウントさせる。

その他の構成と動作は、図 2 2 における場合と同様である。

図 2 8 は、コンバータ 2 0 2 の他の構成例を表している。この構成例においては、入力データのうち、連続する 0 の数をカウンタ 3 7 1 がカウントし、そのカ

ウント結果をコントローラ 3 4 6 に出力するようになされている。カウンタ 3 7 1 のカウント値が 2 2 に達したとき、コントローラ 3 4 6 は、信号 delete position をディリート回路 3 4 3 に出力するとともに、カウンタ 3 7 1 をリセットし、再び新たな連続する 0 のビットの数をカウンタ 3 7 1 にカウントさせる。その他の構成は、図 2 5 における場合と同様である。

このように、この構成例においては、所定のパターン（データ” 0 ”の連続する数）に基づいて、マーカービットとしてのデータ” 1 ”が挿入され、また、削除されることになる。

図 2 7 と図 2 8 に示す構成は、図 2 2 と図 2 5 に示す構成よりも効率的な処理が可能となる。但し、変換後の長さが元の履歴情報に依存することになる。

図 2 9 は、ユーザデータフォーマッタ 2 1 3 の構成例を表している。この例においては、コントローラ 3 8 3 がコンバータ 2 1 2 とユーザデータフォーマッタ 2 1 3 との間に配置されているバッファメモリ（図示せず）に読み出しアドレスを出力すると、そこから読み出されたデータが、ユーザデータフォーマッタ 2 1 3 のスイッチ 3 8 2 の接点 a 側に供給される。ROM 3 8 1 には、ユーザデータスタートコード、データ ID などの user_data() を生成するのに必要なデータが記憶されている。コントローラ 3 1 3 は、所定のタイミングにおいて、スイッチ 3 8 2 を接点 a 側または接点 b 側に切り替え、ROM 3 8 1 に記憶されているデータ、またはコンバータ 2 1 2 より供給されるデータを適宜選択し、出力する。これにより、user_data() のフォーマットのデータが符号化装置 1 0 6 に出力される。

なお、図示は省略するが、ユーザデータデコーダ 2 0 1 は、図 2 9 の ROM 3 8 1 より読み出され、挿入されたデータを削除するスイッチを介して、入力データを出力するようにすることで実現することができる。

図 3 0 は、例えば映像編集スタジオにおいて、複数のトランスコード 1 0 1 - 1 乃至 1 0 1 - N が直列に接続されて使用される状態を示している。各トランスコード 1 0 1 - i (i = 1 乃至 N) の履歴情報多重化装置 1 0 3 - i は、上

述した符号化パラメータ用の領域の最も古い符号化パラメータが記録されている区画に、自己が用いた最新の符号化パラメータを上書きする。このことにより、ベースバンドの画像データには、同一のマクロブロックに対応する直近の4世代分の符号化パラメータ（世代履歴情報）が記録されることになる（図18）。

各符号化装置106-iの符号化装置106-i（図19）は、その可変長符号化回路58において、ヒストリ情報分離装置105-iから供給される今回用いる符号化パラメータに基づいて、量子化回路57より供給されるビデオデータを符号化する。このようにして生成されるビットストリーム（例えば、`picture_header()`）中に、その現符号化パラメータは多重化される。

可変長符号化回路58はまた、ヒストリエンコーディング装置107-iより供給されるユーザデータ（世代履歴情報を含む）を、出力するビットストリーム中に多重化する（図18に示すような埋め込み処理ではなく、ビットストリーム中に多重化する）。そして、符号化装置106-iの出力するビットストリームは、SDTI(Serial Data Transfer Interface)108-iを介して、後段のトランスコード101-(i+1)に入力される。

トランスコード101-iとトランスコード101-(i+1)は、それぞれ図15に示すように構成されている。従って、その処理は、図15を参照して説明した場合と同様となる。

実際の符号化パラメータの履歴を利用した符号化として、現在Iピクチャとして符号化されていたものを、PもしくはBピクチャに変更したい場合、過去の符号化パラメータの履歴を見て、過去にPもしくはBピクチャであった場合を探し、これらの履歴が存在した場合は、その動きベクトルなどのパラメータを利用して、ピクチャタイプを変更する。反対に過去に履歴がない場合は、動き検出を行わないピクチャタイプの変更を断念する。もちろん履歴がない場合であっても、動き検出を行えばピクチャタイプを変更できる。

図18に示すフォーマットの場合、4世代分の符号化パラメータを埋め込むようにしたが、I、P、Bの各ピクチャタイプのパラメータを埋め込むようにする

こともできる。図31は、この場合のフォーマットの例を示している。この例では、同一のマクロブロックが、過去にピクチャタイプの変更を伴って符号化されたときにおける、ピクチャタイプ毎に1世代分の符号化パラメータ（ピクチャ履歴情報）が記録される。したがって、図16に示した復号化装置102、および図19に示した符号化装置106は、現在（最新）、第3世代、第2世代、および第1世代の符号化パラメータの代わりに、Iピクチャ、Pピクチャ、およびBピクチャに対応する1世代分の符号化パラメータを入出力することになる。

また、この例の場合、Cb[1][x]とCr[1][x]の空き領域は利用しないので、Cb[1][x]とCr[1][x]の領域を有さない4:2:0フォーマットの画像データにも本発明を適用することができる。

この例の場合、復号装置102は、符号化パラメータを復号と同時に取り出し、ピクチャタイプを判定して、画像信号のピクチャタイプに対応した場所に符号化パラメータを書き込んで（多重化して）ヒストリ情報分離装置105に出力する。ヒストリ情報分離装置105は、符号化パラメータを分離し、これから符号化したいピクチャタイプと、入力された過去の符号化パラメータを考慮して、ピクチャタイプを変更しながら再符号化を行うことができる。

次に、各トランスコード101において、変更が可能なピクチャタイプを判定する処理について、図32のフローチャートを参照して説明する。なお、トランスコード101におけるピクチャタイプの変更は、過去の動きベクトルを利用するので、この処理は動き検出を行わないで実行されることを前提としている。また、以下に説明する処理は、ヒストリ情報分離装置105により実行される。

ステップS1において、ピクチャタイプ毎に1世代分の符号化パラメータ（ピクチャ履歴情報）がヒストリ情報分離装置105に入力される。

ステップS2において、ヒストリ情報分離装置105は、ピクチャ履歴情報の中に、Bピクチャに変更したときの符号化パラメータが存在するか否かを判定する。ピクチャ履歴情報にBピクチャに変更したときの符号化パラメータが存在すると判定された場合、ステップS3に進む。

ステップS 3において、ヒストリ情報分離装置105は、ピクチャ履歴情報の中に、Pピクチャに変更したときの符号化パラメータが存在するか否かを判定する。ピクチャ履歴情報にPピクチャに変更したときの符号化パラメータが存在すると判定された場合、ステップS 4に進む。

ステップS 4において、ヒストリ情報分離装置105は、変更可能なピクチャタイプがIピクチャ、Pピクチャ、およびBピクチャであると判断する。

ステップS 3において、ピクチャ履歴情報にPピクチャに変更したときの符号化パラメータが存在しないと判定された場合、ステップS 5に進む。

ステップS 5において、ヒストリ情報分離装置105は、変更可能なピクチャタイプがIピクチャ、およびBピクチャであると判断する。さらに、ヒストリ情報分離装置105は、特殊処理（Bピクチャの履歴情報に含まれる後方予測ベクトルを使わず、前方予測ベクトルだけを使う）を施すことにより、擬似的にPピクチャに変更可能であると判断する。

ステップS 2において、ピクチャ履歴情報にBピクチャに変更したときの符号化パラメータが存在しないと判定された場合、ステップS 6に進む。

ステップS 6において、ヒストリ情報分離装置105は、ピクチャ履歴情報にPピクチャに変更したときの符号化パラメータが存在するか否かを判定する。ピクチャ履歴情報にPピクチャに変更したときの符号化パラメータが存在すると判定された場合、ステップS 7に進む。

ステップS 7において、ヒストリ情報分離装置105は、変更可能なピクチャタイプがIピクチャ、およびPピクチャであると判断する。さらに、ヒストリ情報分離装置105は、特殊処理（Pピクチャに履歴情報に含まれる前方予測ベクトルだけを使う）を施すことにより、Bピクチャに変更可能であると判断する。

ステップS 6において、ピクチャ履歴情報にPピクチャに変更したときの符号化パラメータが存在しないと判定された場合、ステップS 8に進む。ステップS 8において、ヒストリ情報分離装置105は、動きベクトルが存在しないので、変更可能なピクチャタイプがIピクチャだけである（IピクチャなのでIピクチャ

ャ以外には変更できない) と判断する。

ステップ S 4, S 5, S 7, S 8 の処理の次にステップ S 9 において、ヒストリ情報分離装置 105 は、変更可能なピクチャタイプを表示装置 (図示せず) に表示してユーザに通知する。

図 3 3 は、ピクチャタイプ変更の例を示している。ピクチャタイプを変更する場合、GOP を構成するフレーム数を変更される。すなわち、この例の場合、 $N=15$ (GOP のフレーム数 $N=15$)、 $M=3$ (GOP 内の I、または P ピクチャの出現周期 $M=3$) のフレームから構成される 4 Mbps の Long GOP (第 1 世代) から、 $N=1$, $M=1$ のフレームで構成される 50 Mbps の Short GOP (第 2 世代) に変換され、再度、 $N=15$, $M=3$ のフレームから構成される 4 Mbps の Long GOP (第 3 世代) に変換されている。なお、図中において破線は、GOP の境界を示している。

第 1 世代から第 2 世代にピクチャタイプが変更される場合において、上述した変更可能ピクチャタイプ判定処理の説明から明らかなように、全てのフレームは、ピクチャタイプを I ピクチャに変更することが可能である。このピクチャタイプ変更のとき、動画像 (第 0 世代) が第 1 世代に変換されたときに演算された全ての動きベクトルは、ピクチャ履歴情報に保存された (残された) 状態となる。次に、再度 Long GOP に変換される (第 2 世代から第 3 世代にピクチャタイプが変更される) 場合、第 0 世代から第 1 世代に変換されたときのピクチャタイプ毎の動きベクトルが保存されているので、これを再利用することにより、画質劣化を抑えて、再度、Long GOP に変換することが可能となる。

図 3 4 は、ピクチャタイプ変更の他の例を示している。この例の場合、 $N=14$, $M=2$ である 4 Mbps の Long GOP (第 1 世代) から、 $N=2$, $M=2$ である 18 Mbps の Short GOP (第 2 世代) に変換され、さらに、 $N=1$, $M=1$ であるフレーム数が 1 の 50 Mbps の Short GOP (第 3 世代) に変換されて、1 Mbps の、フレーム数 N がランダムな GOP (第 4 世代) に変換される。

この例においても、第 0 世代から第 1 世代に変換されたときのピクチャタイプ毎の動きベクトルが、第 3 世代から第 4 世代への変換のときまで保存される。そ

ここで、図34に示すように、複雑にピクチャタイプを変更しても、保存されている符号化パラメータを再利用されることにより、画質劣化を小さく抑えることができる。さらに、保存されている符号化パラメータの量子化スケールを有効に利用すれば画質劣化の少ない符号化を実現できる。

この量子化スケールの再利用について、図35を参照して説明する。図35は、所定のフレームが、第1世代から第4世代まで常に、Iピクチャに変換されており、ビットレートだけが、4Mbps、18Mbps、または50Mbpsに変更されていることを示している。

例えば、第1世代(4Mbps)から第2世代(18Mbps)への変換の際に、ビットレートの高速化に伴って、細かい量子化スケールで再符号化しても画質は向上しない。なぜならば、過去において粗い量子化ステップで量子化されたデータは、復元しないからである。したがって、図35に示すように、途中でビットレートが高速化しても、それに伴って細かい量子化ステップで量子化することは、情報量が増加するだけであって画質の向上には繋がらない。したがって、過去のもっとも粗い(大きい)量子化スケールを維持するように制御すれば、最も無駄が無く、効率的な符号化が可能となる。

なお、第3世代から第4世代への変更時には、ビットレートは、50Mbpsから4Mbpsに低下されているが、この場合にも、過去のもっとも粗い(大きい)量子化スケールが維持される。

上述したように、ビットレートが変更されるときは、過去の量子化スケールの履歴を利用して符号化することは非常に有効である。

この量子化制御処理について、図36のフローチャートを参照して説明する。ステップS11において、ヒストリ情報分離装置105は、入力されたピクチャ履歴情報に、いまから変換するピクチャタイプの符号化パラメータが存在するかどうかを判定する。変換するピクチャタイプの符号化パラメータが存在すると判定された場合、ステップS12に進む。

ステップS12において、ヒストリ情報分離装置105は、ピクチャ履歴情報

の対象となる符号化パラメータから、history_q_scale_code を抽出する。

ステップS 1 3において、ヒストリ情報分離装置 1 0 5 は、送信バッファ 5 9 から量子化回路 5 7 にフィードバックされるバッファ残量に基づいて、feedback_q_scale_code を演算する。

ステップS 1 4において、ヒストリ情報分離装置 1 0 5 は、history_q_scale_code が feedback_q_scale_code よりも大きい（粗い）か否かを判定する。history_q_scale_code が feedback_q_scale_code よりも大きいと判定された場合、ステップS 1 5に進む。

ステップS 1 5において、ヒストリ情報分離装置 1 0 5 は、量子化スケールとして history_q_scale_code を量子化回路 5 7 に出力する。量子化回路 5 7 は、history_q_scale_code を用いて量子化を実行する。

ステップS 1 6において、フレームに含まれる全てのマクロブロックが量子化されたか否かが判定される。全てのマクロブロックがまだ量子化されていないと判定された場合、ステップS 1 2に戻り、ステップS 1 2乃至S 1 6の処理が、全てのマクロブロックが量子化されるまで繰り返される。

ステップS 1 4において、history_q_scale_code が feedback_q_scale_code よりも大きくない（細かい）いと判定された場合、ステップS 1 7に進む。

ステップS 1 7において、ヒストリ情報分離装置 1 0 5 は、量子化スケールとして feedback_q_scale_code を量子化回路 5 7 に出力する。量子化回路 5 7 は、feedback_q_scale_code を用いて量子化を実行する。

ステップS 1 1において、変換するピクチャタイプの符号化パラメータが、ヒストリ情報中に存在しないと判定された場合、ステップS 1 8に進む。

ステップS 1 8において、ヒストリ情報分離装置 1 0 5 は、送信バッファ 5 9 から量子化回路 5 7 にフィードバックされるバッファ残量に基づいて、feedback_q_scale_code を演算する。

ステップS 1 9において、量子化回路 5 7 は、Feedback_q_scale_code を用いて量子化を実行する。

ステップS 20において、フレームに含まれる全てのマクロブロックが量子化されたか否かが判定される。全てのマクロブロックがまだ量子化されていないと判定された場合、ステップS 18に戻り、ステップS 18乃至S 20の処理が、全てのマクロブロックが量子化されるまで繰り返される。

なお、本実施の形態におけるトランスコーダ101の内部においては、上述したように、復号側と符号側が粗結合されており、符号化パラメータを画像データに多重化させて伝送させたが、図37に示すように、復号装置102と符号化装置106を直接接続する（密結合する）ようにしてもよい。

図15において説明したトランスコーダ101は、第1世代から第3世代の過去の符号化パラメータを符号化装置106に供給するために、ベースバンドビデオデータに過去の符号化パラメータを多重化して伝送するようにしていた。しかしながら、本発明においては、ベースバンドビデオデータに過去の符号化パラメータを多重化する技術は必須ではなく、図37に示されたように、ベースバンドビデオデータとは異なる伝送路（たとえばデータ転送バス）を使用して、過去の符号化パラメータを伝送するようにしても良い。

つまり、図37に示した、復号装置102、ヒストリデコーディング装置104、符号化装置106及びヒストリエンコーディング装置107は、図15において説明した復号装置102、ヒストリデコーディング装置104、符号化装置106及びヒストリエンコーディング装置107とまったく同じ機能及び構成を有している。

復号装置102の可変長復号回路112は、第3世代の符号化ストリームST (3rd)のシーケンス層、GOP層、ピクチャ層、スライス層及びマクロブロック層から、第3世代の符号化パラメータを抽出し、それを、ヒストリエンコーディング装置107及び符号化装置106のコントローラ70にそれぞれ供給する。ヒストリエンコーディング装置107は、受け取った第3世代の符号化パラメータをピクチャ層のユーザデータエリアに記述できるように
converted_history_stream()に変換し、converted_history_stream()をユーザデ

ータとして符号化装置 106 の可変長符号化回路 58 に供給する。

さらに可変長復号回路 112 は、第 3 世代の符号化ストリームのピクチャ層のユーザデータエリアから、第 1 世代の符号化パラメータ及び第 2 の符号化パラメータを含んでいるユーザデータ `user_data` を抽出し、ヒストリデコーディング装置 104 及び符号化装置 106 の可変長符号化回路 58 に供給する。ヒストリデコーディング装置 104 は、ユーザデータエリアに

`converted_history_stream()` として記述されたヒストリストリームから、第 1 世代の符号化パラメータ及び第 2 世代の符号化パラメータを抽出し、それを符号化装置 106 のコントローラに供給する。

符号化装置 106 のコントローラ 70 は、ヒストリデコーディング装置 104 から受け取った第 1 世代及び第 2 世代の符号化パラメータと、符号化装置 102 から受け取った第 3 世代の符号化パラメータとに基づいて、符号化装置 106 の符号化処理をコントロールする。

符号化装置 106 の可変長符号化回路 58 は、復号装置 102 から第 1 世代の符号化パラメータ及び第 2 の符号化パラメータを含んでいるユーザデータ `user_data` を受け取るとともに、ヒストリエンコーディング装置 107 から第 3 世代の符号化パラメータを含んでいるユーザデータ `user_data` を受け取り、それらのユーザデータをヒストリ情報として、第 4 世代の符号化ストリームのピクチャ層のユーザデータエリアに記述する。

図 38 は、MPEG のビデオストリームをデコードするためのシンタックスを表わした図である。デコードは、このシンタックスに従って MPEG ビットストリームをデコードすることによって、ビットストリームから意味のある複数のデータ項目（データエレメント）を抽出する。以下に説明するシンタックスは、図において、その関数や条件文は細活字で表わされ、そのデータエレメントは、太活字で表されている。データ項目は、その名称、ビット長、及びそのタイプと伝送順序を示すニーモニック（Mnemonic）で記述されている。

まず、この図 38 に示されているシンタックスにおいて使用されている関数に

について説明する。

`next_start_code()` 関数は、ビットストリーム中に記述されているスタートコードを探すための関数である。この図 3 8 に示されたシンタックスにおいて、この `next_start_code()` 関数の次に、`sequence_header()` 関数と `sequence_extension()` 関数とが順に配置されているので、このビットストリームには、この `sequence_header()` 関数と `sequence_extension()` 関数によって定義されたデータエレメントが記述されている。従って、ビットストリームのデコード時には、この `next_start_code()` 関数によって、`sequence_header()` 関数と `sequence_extension()` 関数の先頭に記述されているスタートコード（データエレメントの一種）をビットストリーム中から見つけ、それを基準にして、

`sequence_header()` 関数と `sequence_extension()` 関数をさらに見つけ、それらによって定義された各データエレメントをデコードする。

尚、`sequence_header()` 関数は、MPEG ビットストリームのシーケンス層のヘッダデータを定義するための関数であって、`sequence_extension()` 関数は、MPEG ビットストリームのシーケンス層の拡張データを定義するための関数である。

`sequence_extension()` 関数の次に配置されている `do{ }while` 構文は、`while` 文によって定義されている条件が真である間、`do` 文の `{ }` 内の関数に基いて記述されたデータエレメントをデータストリーム中から抽出するための構文である。すなわち、

`do{ }while` 構文によって、`while` 文によって定義されている条件が真である間、ビットストリーム中から、`do` 文内の関数に基いて記述されたデータエレメントを抽出するデコード処理が行われる。

この `while` 文に使用されている `nextbits()` 関数は、ビットストリーム中に現れるビット又はビット列と、次にデコードされるデータエレメントとを比較するための関数である。この図 3 8 のシンタックスの例では、`nextbits()` 関数は、ビットストリーム中のビット列とビデオシーケンスの終わりを示す `sequence_end_code` とを比較し、ビットストリーム中のビット列と

sequence_end_code とが一致しないときに、この while 文の条件が真となる。従って、sequence_extension() 関数の次に配置されている do{ }while 構文は、ビットストリーム中に、ビデオシーケンスの終わりを示す sequence_end_code が現れない間、do 文中の関数によって定義されたデータエレメントがビットストリーム中に記述されていることを示している。

ビットストリーム中には、sequence_extension() 関数によって定義された各データエレメントの次には、extension_and_user_data(0) 関数によって定義されたデータエレメントが記述されている。この extension_and_user_data(0) 関数は、MPEG ビットストリームのシーケンス層の拡張データとユーザデータを定義するための関数である。

この extension_and_user_data(0) 関数の次に配置されている do{ }while 構文は、while 文によって定義されている条件が真である間、do 文の{ }内の関数に基づいて記述されたデータエレメントを、ビットストリーム中から抽出するための関数である。この while 文において使用されている nextbits() 関数は、ビットストリーム中に現れるビット又はビット列と、picture_start_code 又は group_start_code との一致を判断するための関数であって、ビットストリーム中に現れるビット又はビット列と、picture_start_code 又は group_start_code とが一致する場合には、while 文によって定義された条件が真となる。よって、この do{ }while 構文は、ビットストリーム中において、picture_start_code 又は group_start_code が現れた場合には、そのスタートコードの次に、do 文中の関数によって定義されたデータエレメントのコードが記述されているので、この picture_start_code 又は group_start_code によって示されるスタートコードを探し出すことによって、ビットストリーム中から do 文中に定義されたデータエレメントを抽出することができる。

この do 文の最初に記述されている if 文は、ビットストリーム中に group_start_code が現れた場合、という条件を示している。この if 文による条件が真である場合には、ビットストリーム中には、この group_start_code の次に

group_of_picture_header(1)関数及び extension_and_user_data(1)関数によって定義されているデータエレメントが順に記述されている。

この group_of_picture_header(1)関数は、MPEG ビットストリームの GOP 層のヘッダデータを定義するための関数であって、

extension_and_user_data(1)関数は、MPEG ビットストリームの GOP 層の拡張データ (extension_data) 及びユーザデータ (user_data) を定義するための関数である。

さらに、このビットストリーム中には、group_of_picture_header(1)関数及び extension_and_user_data(1)関数によって定義されているデータエレメントの次に、picture_header()関数と picture_coding_extension()関数によって定義されたデータエレメントが記述されている。もちろん、先に説明した if 文の条件が真とならない場合には、

group_of_picture_header(1)関数及び extension_and_user_data(1)関数によって定義されているデータエレメントは記述されていないので、

extension_and_user_data(0)関数によって定義されているデータエレメントの次に、

picture_header()関数と picture_coding_extension()関数によって定義されたデータエレメントが記述されている。

この picture_header()関数は、

MPEG ビットストリームのピクチャ層のヘッダデータを定義するための関数であって、

picture_coding_extension()関数は、MPEG ビットストリームのピクチャ層の第 1 の拡張データを定義するための関数である。

次の while 文は、この while 文によって定義されている条件が真である間、次の if 文の条件判断を行うための関数である。この while 文において使用されている nextbits()関数は、ビットストリーム中に現れるビット列と、extension_start_code 又は user_data_start_code との一致を判断するための関

数であって、ビットストリーム中に現れるビット列と、

`extension_start_code` 又は `user_data_start_code` とが一致する場合には、この `while` 文によって定義された条件が真となる。

第1の `if` 文は、ビットストリーム中に現れるビット列と `extension_start_code` との一致を判断するための関数である。ビットストリーム中に現れるビット列と 32 ビットの `extension_start_code` とが一致する場合には、ビットストリーム中において、`extension_start_code` の次に `extension_data(2)` 関数によって定義されるデータエレメントが記述されている。

第2の `if` 文は、ビットストリーム中に現れるビット列と `user_data_start_code` との一致を判断するための構文であって、ビットストリーム中に現れるビット列と 32 ビットの `user_data_start_code` とが一致する場合には、第3の `if` 文の条件判断が行われる。この `user_data_start_code` は、MPEG ビットストリームのピクチャ層のユーザデータエリアの開始を示すためのスタートコードである。

第3の `if` 文は、ビットストリーム中に現れるビット列と `History_Data_ID` との一致を判断するための構文である。ビットストリーム中に現れるビット列とこの 32 ビットの `History_Data_ID` とが一致する場合には、この MPEG ビットストリームのピクチャ層のユーザデータエリアにおいて、この 32 ビットの `History_Data_ID` によって示されるコードの次に、`converted_history_stream()` 関数によって定義されるデータエレメントが記述されている。

`converted_history_stream()` 関数は、MPEG 符号化時に使用したあらゆる符号化パラメータを伝送するための履歴情報及び履歴データを記述するための関数である。この `converted_history_stream()` 関数によって定義されているデータエレメントの詳細は、図 40 乃至図 47 を参照して、`history_stream()` として後述する。また、この `History_Data_ID` は、MPEG ビットストリームのピクチャ層のユーザデータエリアに記述されたこの履歴情報及び履歴データが記述されている。

先頭を示すためのスタートコードである。

else 文は、第3の if 文において、条件が非真であることを示すための構文である。従って、この MPEG ビットストリームのピクチャ層のユーザデータエリアにおいて、converted_history_stream() 関数によって定義されたデータエレメントが記述されていない場合には、user_data() 関数によって定義されたデータエレメントが記述されている。

図38において、履歴情報は、converted_history_stream() に記述され、user_data() に記述される訳ではないが、この converted_history_stream() は、MPEG 規格の user_data の一種として記述される。そこで、本明細書中においては、場合によって、履歴情報が user_data に記述されるとも説明するが、それは、MPEG 規格の user_data の一種として記述されるということを意味する。

picture_data() 関数は、MPEG ビットストリームのピクチャ層のユーザデータの次に、スライス層及びマクロブロック層に関するデータエレメントを記述するための関数である。通常は、この picture_data() 関数によって示されるデータエレメントは、ビットストリームのピクチャ層のユーザデータエリアに記述された converted_history_stream() 関数によって定義されるデータエレメント又は user_data() 関数によって定義されたデータエレメントの次に記述されているが、ピクチャ層のデータエレメントを示すビットストリーム中に、extension_start_code 又は user_data_start_code が存在しない場合には、この picture_data() 関数によって示されるデータエレメントは、

picture_coding_extension() 関数によって定義されるデータエレメントの次に記述されている。

この picture_data() 関数によって示されるデータエレメントの次には、sequence_header() 関数と sequence_extension() 関数とによって定義されたデータエレメントが順に配置されている。この sequence_header() 関数と sequence_extension() 関数によって記述されたデータエレメントは、ビデオストリームのシーケンスの先頭に記述された sequence_header() 関数と

sequence_extension()関数によって記述されたデータエレメントと全く同じである。このように同じデータをストリーム中に記述する理由は、ビットストリーム受信装置側でデータストリームの途中（例えばピクチャ層に対応するビットストリーム部分）から受信が開始された場合に、シーケンス層のデータを受信できなくなり、ストリームをデコード出来なくなることを防止するためである。

この最後の sequence_header()関数と sequence_extension()関数とによって定義されたデータエレメントの次、つまり、データストリームの最後には、シーケンスの終わりを示す32ビットの sequence_end_code が記述されている。

以上のシンタックスの基本的な構成の概略を示すと、図39に示すようになる。

次に、converted_history_stream()関数によって定義されたヒストリストリームに関して説明する。

この converted_history_stream()は、MPEGのピクチャ層のユーザデータエリアに履歴情報を示すヒストリストリームを挿入するための関数である。尚、

「converted」の意味は、スタートエミュレーションを防止するために、ユーザエリアに挿入すべき履歴データから構成される履歴ストリームの少なくとも22ビット毎にマーカービット（1ビット）を挿入する変換処理を行ったストリームであることを意味している。

この converted_history_stream()は、以下に説明する固定長の履歴ストリーム（図40乃至図46）又は可変長の履歴ストリーム（図47）のいずれかの形式で記述される。エンコーダ側において固定長の履歴ストリームを選択した場合には、デコーダ側において履歴ストリームから各データエレメントをデコードするための回路及びソフトウェアが簡単になるというメリットがある。一方、エンコーダ側において可変長の履歴ストリームを選択した場合には、エンコーダにおいてピクチャ層のユーザエリアに記述される履歴情報（データエレメント）を必要に応じて任意に選択することができるので、履歴ストリームのデータ量を少なくすることができ、その結果、符号化されたビットストリーム全体のデータレ

トを低減することができる。

本発明において説明する「履歴ストリーム」、「ヒストリストリーム」、「履歴情報」、「ヒストリ情報」、「履歴データ」、「ヒストリデータ」、「履歴パラメータ」、「ヒストリパラメータ」とは、過去の符号化処理において使用した符号化パラメータ（又はデータエレメント）を意味し、現在の（最終段の）符号化処理において使用した符号化パラメータを意味するものではない。例えば、第1世代の符号化処理において、あるピクチャをIピクチャで符号化して伝送し、次なる第2世代の符号化処理において、このピクチャを今度はPピクチャとして符号化して伝送し、さらに、第3世代の符号化処理において、このピクチャをBピクチャで符号化して伝送する例をあげて説明する。

第3世代の符号化処理において使用した符号化パラメータが、第3世代の符号化処理において生成された符号化ビットストリームのシーケンス層、GOP層、ピクチャ層、スライス層及びマクロブロック層の所定位置に記述されている。一方、過去の符号化処理である第1世代及び第2世代の符号化処理において使用した符号化パラメータは、第3世代の符号化処理において使用した符号化パラメータが記述されるシーケンス層やGOP層に記述されるのではなく、既に説明したシンタックスに従って、符号化パラメータの履歴情報として、ピクチャ層のユーザデータエリアに記述される。

まず、固定長の履歴ストリームシンタックスについて図40乃至図46を参照して説明する。

最終段（例えば第3世代）の符号化処理において生成されたビットストリームのピクチャ層のユーザデータエリアには、まず最初に、過去（例えば第1世代及び第2世代）の符号化処理において使用されていたシーケンス層のシーケンスヘッダに含められる符号化パラメータが、履歴ストリームとして挿入される。尚、過去の符号化処理において生成されたビットストリームのシーケンス層のシーケンスヘッダ等の履歴情報は、最終段の符号化処理において生成されたビットストリームのシーケンス層のシーケンスヘッダに挿入されることは無いという点に注

意すべきである。

過去の符号化処理で使用したシーケンスヘッダ (sequence_header) に含まれるデータエレメントは、sequence_header_code、sequence_header_present_flag、horizontal_size_value、marker_bit、vertical_size_value、aspect_ratio_information、frame_rate_code、bit_rate_value、VBV_buffer_size_value、constrained_parameter_flag、load_intra_quantiser_matrix、load_non_intra_quantiser_matrix、intra_quantiser_matrix、及び non_intra_quantiser_matrix 等から構成される。

sequence_header_code は、シーケンス層のスタート同期コードを表すデータである。sequence_header_present_flag は、sequence_header 内のデータが有効か無効かを示すデータである。horizontal_size_value は、画像の水平方向の画素数の下位 12 ビットから成るデータである。marker_bit は、スタートコードエミュレーションを防止するために挿入されるビットデータである。vertical_size_value は、画像の縦のライン数の下位 12 ビットからなるデータである。aspect_ratio_information は、画素のアスペクト比（縦横比）または表示画面アスペクト比を表すデータである。frame_rate_code は、画像の表示周期を表すデータである。

bit_rate_value は、発生ビット量に対する制限のためのビット・レートの下位 18 ビット (400bsp 単位で切り上げる) データである。VBV_buffer_size_value は、発生符号量制御用の仮想バッファ（ビデオバッファベリファイヤー）の大きさを決める値の下位 10 ビットデータである。constrained_parameter_flag は、各パラメータが制限以内であることを示すデータである。

load_intra_quantiser_matrix は、イントラ MB 用量子化マトリックス・データの存在を示すデータである。load_non_intra_quantiser_matrix は、非イントラ MB 用量子化マトリックス・データの存在を示すデータである。

intra_quantiser_matrix は、イントラ MB 用量子化マトリックスの値を示すデ

ータである。non_intra_quantiser_matrix は、非イントラMB用量子化マトリックスの値を表すデータである。

最終段の符号化処理において生成されたビットストリームのピクチャ層のユーザデータエリアには、過去の符号化処理において使用されたシーケンス層のシーケンスエクステンションを表わすデータエレメントが、履歴ストリームとして記述される。

この過去の符号化処理で使用したシーケンスエクステンション (sequence_extension) を表わすデータエレメントは、

extension_start_code、extension_start_code_identifier、sequence_extension_present_flag、profile_and_level_indication、progressive_sequence、chroma_format、horizontal_size_extension、vertical_size_extension、bit_rate_extension、vbv_buffer_size_extension、low_delay、frame_rate_extension_n、及び frame_rate_extension_d 等のデータエレメントである。

extension_start_code は、エクステンションデータのスタート同期コードを表すデータである。extension_start_code_identifier は、どの拡張データが送られるかを示すデータである。sequence_extension_present_flag は、シーケンスエクステンション内のデータが有効であるか無効であるかを示すデータである。profile_and_level_indication は、ビデオデータのプロファイルとレベルを指定するためのデータである。progressive_sequence は、ビデオデータが順次走査であることを示すデータである。chroma_format は、ビデオデータの色差フォーマットを指定するためのデータである。

horizontal_size_extension は、シーケンスヘッダの horizontal_size_value に加える上位2ビットのデータである。vertical_size_extension は、シーケンスヘッダの vertical_size_value に加える上位2ビットのデータである。bit_rate_extension は、シーケンスヘッダの bit_rate_value に加える上位12ビットのデータである。vbv_buffer_size_extension は、シーケンスヘッダの

vbv_buffer_size_value に加える上位 8 ビットのデータである。low_delay は、B ピクチャを含まないことを示すデータである。frame_rate_extension_n は、シーケンスヘッダの frame_rate_code と組み合わせてフレームレートを得るためのデータである。frame_rate_extension_d は、シーケンスヘッダの frame_rate_code と組み合わせてフレームレートを得るためのデータである。

続いて、ビットストリームのピクチャ層のユーザエリアには、過去の符号化処理において使用されたシーケンス層のシーケンスディスプレイエクステンションを表わすデータエレメントが、履歴ストリームとして記述される。

このシーケンスディスプレイエクステンション

(sequence_display_extension) として記述されているデータエレメントは、extension_start_code、extension_start_code_identifier、sequence_display_extension_present_flag、video_format、colour_description、colour_primaries、transfer_characteristics、matrix_coefficients、display_horizontal_size、及び display_vertical_size から構成される。

extension_start_code は、エクステンションデータのスタート同期コードを表すデータである。extension_start_code_identifier は、どの拡張データが送られるかを示すコードである。sequence_display_extension_present_flag は、シーケンスディスプレイエクステンション内のデータエレメントが有効か無効かを示すデータである。video_format は、原信号の映像フォーマットを表すデータである。color_description は、色空間の詳細データがあることを示すデータである。color_primaries は、原信号の色特性の詳細を示すデータである。transfer_characteristics は、光電変換がどのように行われたのかの詳細を示すデータである。matrix_coefficients は、原信号が光の三原色からどのように変換されたかの詳細を示すデータである。display_horizontal_size は、意図するディスプレイの活性領域（水平サイズ）を表すデータである。display_vertical_size は、意図するディスプレイの活性領域（垂直サイズ）を

表すデータである。

続いて、最終段の符号化処理において生成されたビットストリームのピクチャ層のユーザエリアには、過去の符号化処理において生成されたマクロブロックの位相情報を示すマクロブロックアサイメントデータ

(macroblock_assignment_in_user_data) が、履歴ストリームとして記述される。

このマクロブロックの位相情報を示す macroblock_assignment_in_user_data は、macroblock_assignment_present_flag、v_phase、h_phase 等のデータエレメントから構成される。

この macroblock_assignment_present_flag は、macroblock_assignment_in_user_data 内のデータエレメントが有効か無効かを示すデータである。v_phase は、画像データからマクロブロックを切り出す際の垂直方向の位相情報を示すデータである。h_phase は、画像データからマクロブロックを切り出す際の水平方向の位相情報を示すデータである。

続いて、最終段の符号化処理によって生成されたビットストリームのピクチャ層のユーザエリアには、過去の符号化処理において使用された GOP 層の GOP ヘッダを表わすデータエレメントが、履歴ストリームとして記述されている。

この GOP ヘッダ (group_of_picture_header) を表わすデータエレメントは、group_start_code、group_of_picture_header_present_flag、time_code、closed_gop、及び broken_link から構成される。

group_start_code は、GOP 層の開始同期コードを示すデータである。

group_of_picture_header_present_flag は、

group_of_picture_header 内のデータエレメントが有効であるか無効であるかを示すデータである。time_code は、GOP の先頭ピクチャのシーケンスの先頭からの時間を示すタイムコードである。closed_gop は、GOP 内の画像が他の GOP から独立再生可能なことを示すフラグデータである。broken_link は、編集などのために GOP 内の先頭の B ピクチャが正確に再生できないことを示すフラグデータ

である。

続いて、最終段の符号化処理によって生成されたビットストリームのピクチャ層のユーザエリアには、過去の符号化処理において使用されたピクチャ層のピクチャヘッダを表わすデータエレメントが、履歴ストリームとして記述されている。

このピクチャヘッダ (picture_header) に関するデータエレメントは、picture_start_code、temporal_reference、picture_coding_type、vbm_delay、full_pel_forward_vector、forward_f_code、full_pel_backward_vector、及び backward_f_code から構成される。

具体的には、picture_start_code は、ピクチャ層の開始同期コードを表すデータである。temporal_reference は、ピクチャの表示順を示す番号で GOP の先頭でリセットされるデータである。picture_coding_type は、ピクチャタイプを示すデータである。vbm_delay は、ランダムアクセス時の仮想バッファの初期状態を示すデータである。full_pel_forward_vector は、順方向動きベクトルの精度が整数単位か半画素単位かを示すデータである。forward_f_code は、順方向動きベクトル探索範囲を表すデータである。full_pel_backward_vector は、逆方向動きベクトルの精度が整数単位か半画素単位かを示すデータである。backward_f_code は、逆方向動きベクトル探索範囲を表すデータである。

続いて、最終段の符号化処理によって生成されたビットストリームのピクチャ層のユーザエリアには、過去の符号化処理において使用されたピクチャ層のピクチャコーディングエクステンションが、履歴ストリームとして記述されている。

このピクチャコーディングエクステンション (picture_coding_extension) に関するデータエレメントは、extension_start_code、extension_start_code_identifier、f_code[0][0]、f_code[0][1]、f_code[1][0]、f_code[1][1]、intra_dc_precision、picture_structure、top_field_first、frame_predictive_frame_dct、concealment_motion_vectors、q_scale_type、intra_vlc_format、alternate_scan、repeat_firt_field、

chroma_420_type、progressive_frame、composite_display_flag、v_axis、field_sequence、sub_carrier、burst_amplitude、及び sub_carrier_phase から構成される。

extension_start_code は、ピクチャ層のエクステンションデータのスタートを示す開始コードである。extension_start_code_identifier は、どの拡張データが送られるかを示すコードである。f_code[0][0]は、フォワード方向の水平動きベクトル探索範囲を表すデータである。f_code[0][1]は、フォワード方向の垂直動きベクトル探索範囲を表すデータである。f_code[1][0]は、バックワード方向の水平動きベクトル探索範囲を表すデータである。f_code[1][1]は、バックワード方向の垂直動きベクトル探索範囲を表すデータである。

intra_dc_precision は、DC 係数の精度を表すデータである。

picture_structure は、フレームストラクチャかフィールドストラクチャかを示すデータである。フィールドストラクチャの場合は、上位フィールドか下位フィールドかもあわせて示すデータである。top_field_first は、フレームストラクチャの場合、最初のフィールドが上位か下位かを示すデータである。

frame_predictive_frame_dct は、フレーム・ストラクチャの場合、フレーム・モード DCT の予測がフレーム・モードだけであることを示すデータである。

concealment_motion_vectors は、イントラマクロブロックに伝送エラーを隠蔽するための動きベクトルがついていることを示すデータである。

q_scale_type は、線形量子化スケールを利用するか、非線形量子化スケールを利用するかを示すデータである。intra_vlc_format は、イントラマクロブロックに、別の 2 次元 VLC を使うかどうかを示すデータである。alternate_scan は、ジグザグスキャンを使うか、オルタネート・スキャンを使うかの選択を表すデータである。repeat_firt_field は、2 : 3 プルダウンの際に使われるデータである。chroma_420_type は、信号フォーマットが 4 : 2 : 0 の場合、次の progressive_frame と同じ値、そうでない場合は 0 を表すデータである。progressive_frame は、このピクチャが、順次走査できているかどうかを示すデ

ータである。composite_display_flag は、ソース信号がコンポジット信号であったかどうかを示すデータである。

v_axis は、ソース信号が、PAL の場合に使われるデータである。

field_sequence は、ソース信号が、PAL の場合に使われるデータである。

sub_carrier は、ソース信号が、PAL の場合に使われるデータである。

burst_amplitude は、ソース信号が、PAL の場合に使われるデータである。

sub_carrier_phase は、ソース信号が、PAL の場合に使われるデータである。

続いて、最終段の符号化処理によって生成されたビットストリームのピクチャ層のユーザエリアには、過去の符号化処理において使用された量子化マトリックスエクステンションが、履歴ストリームとして記述されている。

この量子化マトリックスエクステンション (quant_matrix_extension) に関するデータエレメントは、extension_start_code、

extension_start_code_identifier、quant_matrix_extension_present_flag、

load_intra_quantiser_matrix、intra_quantiser_matrix[64]、

load_non_intra_quantiser_matrix、non_intra_quantiser_matrix[64]、

load_chroma_intra_quantiser_matrix、chroma_intra_quantiser_matrix[64]、

load_chroma_non_intra_quantiser_matrix、及び

chroma_non_intra_quantiser_matrix[64] から構成される。

extension_start_code は、この量子化マトリックスエクステンションのスタートを示す開始コードである。extension_start_code_identifier は、どの拡張データが送られるかを示すコードである。

quant_matrix_extension_present_flag は、この量子化マトリックスエクステンション内のデータエレメントが有効か無効かを示すためのデータである。

load_intra_quantiser_matrix は、イントラマクロブロック用の量子化マトリックスデータの存在を示すデータである。intra_quantiser_matrix は、イントラマクロブロック用の量子化マトリックスの値を示すデータである。

load_non_intra_quantiser_matrix は、非イントラマクロブロック用の量子化

マトリックスデータの存在を示すデータである。non_intra_quantiser_matrix は、非イントラマクロブロック用の量子化マトリックスの値を表すデータである。load_chroma_intra_quantiser_matrix は、色差イントラマクロブロック用の量子化マトリックス・データの存在を示すデータである。

chroma_intra_quantiser_matrix は、色差イントラマクロブロック用の量子化マトリックスの値を示すデータである。load_chroma_non_intra_quantiser_matrix は、色差非イントラマクロブロック用の量子化マトリックス・データの存在を示すデータである。chroma_non_intra_quantiser_matrix は、色差非イントラマクロブロック用の量子化マトリックスの値を示すデータである。

続いて、最終段の符号化処理によって生成されたビットストリームのピクチャ層のユーザエリアには、過去の符号化処理において使用されたコピーライトエクステンションが、履歴ストリームとして記述されている。

このコピーライトエクステンション (copyright_extension) に関するデータエレメントは、extension_start_code、extension_start_code_identifier、copyright_extension_present_flag、copyright_flag、copyright_identifier、original_or_copy、copyright_number_1、copyright_number_2、及び copyright_number_3 から構成される。

extension_start_code は、コピーライトエクステンションのスタートを示す開始コードである。extension_start_code_identifier のどのエクステンションデータが送られるかを示すコードである。copyright_extension_present_flag は、このコピーライトエクステンション内のデータエレメントが有効か無効かを示すためのデータである。copyright_flag は、次のコピーライトエクステンション又はシーケンスエンドまで、符号化されたビデオデータに対してコピー権が与えられているか否かを示す。

copyright_identifier は、ISO/IEC JTC/SC29 によって指定されたコピー権の登録機関を識別するためのデータである。original_or_copy は、ビットストリーム中のデータが、オリジナルデータであるかコピーデータであるかを示すデー

タである。copyright_number_1 は、コピーライトナンバーのビット 4 4 から 6 3 を表わすデータである。copyright_number_2 は、コピーライトナンバーのビット 2 2 から 4 3 を表わすデータである。copyright_number_3 は、コピーライトナンバーのビット 0 から 2 1 を表わすデータである。

続いて、最終段の符号化処理によって生成されたビットストリームのピクチャ層のユーザエリアには、過去の符号化処理において使用されたピクチャディスプレイエクステンション (picture_display_extension) が、履歴ストリームとして記述されている。

このピクチャディスプレイエクステンションを表わすデータエレメントは、extension_start_code、extension_start_code_identifier、picture_display_extension_present_flag、frame_center_horizontal_offset_1、frame_center_vertical_offset_1、frame_center_horizontal_offset_2、frame_center_vertical_offset_2、frame_center_horizontal_offset_3、及び frame_center_vertical_offset_3 から構成される。

extension_start_code は、ピクチャディスプレイエクステンションのスタートを示すための開始コードである。extension_start_code_identifier は、どの拡張データが送られるかを示すコードである。

picture_display_extension_present_flag は、ピクチャディスプレイエクステンション内のデータエレメントが有効か無効かを示すデータである。

frame_center_horizontal_offset は、表示エリアの水平方向のオフセットを示すデータであって、3つのオフセット値まで定義することができる。

frame_center_vertical_offset は、表示エリアを垂直方向のオフセットを示すデータであって、3つのオフセット値まで定義することができる。

本発明の特徴的な点として、このピクチャディスプレイエクステンションを表わす履歴情報の次に、re_coding_stream_information に関するデータエレメントが記述されている。re_coding_stream_information は re_coding_stream_information に関するデータエレメントは、

user_data_start_code, re_coding_stream_info_ID, red_bw_flag, red_bw_indicator などのデータエレメントより構成される。

user_data_start_code は、user_data が開始することを表すスタートコードである。re_coding_stream_info_ID は、16 ビットの整数であり、re_coding_stream_info() 関数の識別のために用いられる。その値は、具体的には、"1001 0001 1110 1100" (0x91ec) とされる。

red_bw_flag は、1 ビットのフラグであり、全ての履歴情報に関するコーディングパラメータを伝送する場合には0とされ、履歴情報に関するコーディングパラメータが選択的に伝送する場合には1とされる。red_bw_indicator は、2 ビットの整数であり、コーディングパラメータのデータセットを定義するためのインジケータである。

この re_coding_stream_information、red_bw_flag、red_bw_indicator 及びデータセットに関しては、詳しくは後述する。

最終段の符号化処理において生成されたビットストリームのピクチャ層のユーザエリアには、過去の符号化処理において使用されたユーザデータ (user_data) が、履歴ストリームとして記述されている。

このユーザデータの次には、過去の符号化処理において使用されたマクロブロック層に関する情報が、履歴ストリームとして記述されている。

このマクロブロック層に関する情報は、macroblock_address_h、macroblock_address_v、slice_header_present_flag、skipped_macroblock_flag 等のマクロブロック (macroblock) の位置に関するデータエレメントと、macroblock_quant、macroblock_motion_forward、macroblock_motion_backward、macroblock_pattern、macroblock_intra、spatial_temporal_weight_code_flag、frame_motion_type、及び dct_type 等のマクロブロックモード (macroblock_modes[]) に関するデータエレメントと、quantiser_scale_code 等の量子化ステップ制御に関するデータエレメントと、PMV[0][0][0]、PMV[0][0][1]、motion_vertical_field_select[0][0]、

PMV[0][1][0]、PMV[0][1][1]、motion_vertical_field_select[0][1]、PMV[1][0][0]、PMV[1][0][1]、motion_vertical_field_select[1][0]、PMV[1][1][0]、PMV[1][1][1]、motion_vertical_field_select[1][1]等の動き補償に関するデータエレメントと、coded_block_pattern等のマクロブロックパターンに関するデータエレメントと、num_mv_bits、num_coef_bits、及びnum_other_bits等の発生符号量に関するデータエレメントから構成されている。

以下にマクロブロック層に関するデータエレメントについて詳細に説明する。

macroblock_address_hは、現在のマクロブロックの水平方向の絶対位置を定義するためのデータである。macroblock_address_vは、現在のマクロブロックの垂直方向の絶対位置を定義するためのデータである。

slice_header_present_flagは、このマクロブロックがスライス層の先頭であり、スライスヘッダを伴うか否かを示すデータである。

skipped_macroblock_flagは、復号処理においてこのマクロブロックをスキップするか否かを示すデータである。

macroblock_quantは、後述する図63と図64に示されたマクロブロックタイプ(macroblock_type)から導かれるデータであって、

quantiser_scale_codeがビットストリーム中に現れるか否かを示すデータである。macroblock_motion_forwardは、図63と図64に示されたマクロブロックタイプから導かれるデータであって、復号処理で使用されるデータである。

macroblock_motion_backwardは、図63と図64に示されたマクロブロックタイプから導かれるデータであって、復号処理で使用されるデータである。

macroblock_patternは、図63と図64に示されたマクロブロックタイプから導かれるデータであって、coded_block_patternがビットストリーム中に現れるか否かを示すデータである。

macroblock_intraは、図63と図64に示されたマクロブロックタイプから導かれるデータであって、復号処理で使用されるデータである。

spatial_temporal_weight_code_flag は、図 6 3 と図 6 4 に示されたマクロブロックタイプから導かれるデータであって、時間スケラビリティで下位レイヤ画像のアップサンプリング方法を示す spatial_temporal_weight_code は、ビットストリーム中に存在するか否かを示すデータである。

frame_motion_type は、フレームのマクロブロックの予測タイプを示す 2 ビットのコードである。予測ベクトルが 2 個でフィールドベースの予測タイプであれば「00」であって、予測ベクトルが 1 個でフィールドベースの予測タイプであれば「01」であって、予測ベクトルが 1 個でフレームベースの予測タイプであれば「10」であって、予測ベクトルが 1 個でディアルプライムの予測タイプであれば「11」である。field_motion_type は、フィールドのマクロブロックの動き予測を示す 2 ビットのコードである。予測ベクトルが 1 個でフィールドベースの予測タイプであれば「01」であって、予測ベクトルが 2 個で 18×8 マクロブロックベースの予測タイプであれば「10」であって、予測ベクトルが 1 個でディアルプライムの予測タイプであれば「11」である。dct_type は、DCT がフレーム DCT モードか、フィールド DCT モードかを示すデータである。

quantiser_scale_code はマクロブロックの量子化ステップサイズを示すデータである。

次に動きベクトルに関するデータエレメントについて説明する。動きベクトルは、復号時に必要な動きベクトルを減少させるために、先に符号化されたベクトルに関し差分として符号化される。動きベクトルの復号を行うために復号器は、4 個の動きベクトル予測値（それぞれ水平及び垂直成分を伴う）を維持しなければならない。この予測動きベクトルを PMV[r][s][v] と表わすことにしている。[r] は、マクロブロックにおける動きベクトルが第 1 のベクトルであるのか、第 2 のベクトルであるのかを示すフラグであって、マクロブロックにおけるベクトルが第 1 のベクトルである場合には「0」となって、マクロブロックにおけるベクトルが第 2 のベクトルである場合には「1」となる。[s] は、マクロブロックにおける動きベクトルの方向が、前方向であるのか後方向であるのかを示すフ

ラグであって、前方向動きベクトルの場合には「0」となって、後方向動きベクトルの場合には「1」となる。 $[v]$ は、マクロブロックにおけるベクトルの成分が、水平方向であるのか垂直方向であるのかを示すフラグであって、水平方向成分の場合には「0」となって、垂直方向成分の場合には「1」となる。

従って、 $PMV[0][0][0]$ は、第1のベクトルの前方向の動きベクトルの水平方向成分のデータを表わし、 $PMV[0][0][1]$ は、第1のベクトルの前方向の動きベクトルの垂直方向成分のデータを表わし、 $PMV[0][1][0]$ は、第1のベクトルの後方向の動きベクトルの水平方向成分のデータを表わし、 $PMV[0][1][1]$ は、第1のベクトルの後方向の動きベクトルの垂直方向成分のデータを表わし、

$PMV[1][0][0]$ は、第2のベクトルの前方向の動きベクトルの水平方向成分のデータを表わし、 $PMV[1][0][1]$ は、第2のベクトルの前方向の動きベクトルの垂直方向成分のデータを表わし、

$PMV[1][1][0]$ は、第2のベクトルの後方向の動きベクトルの水平方向成分のデータを表わし、 $PMV[1][1][1]$ は、第2のベクトルの後方向の動きベクトルの垂直方向成分のデータを表わしている。

$motion_vertical_field_select[r][s]$ は、予測の形式にいずれの参照フィールドを使用するかを示すデータである。この $motion_vertical_field_select[r][s]$ が「0」の場合には、トップ参照フィールドを使用し、「1」の場合には、ボトム参照フィールドを使用することを示している。

よって、 $motion_vertical_field_select[0][0]$ は、第1のベクトルの前方向の動きベクトルを生成する際の参照フィールドを示し、 $motion_vertical_field_select[0][1]$ は、第1のベクトルの後方向の動きベクトルを生成する際の参照フィールドを示し、 $motion_vertical_field_select[1][0]$ は、第2のベクトルの前方向の動きベクトルを生成する際の参照フィールドを示し、 $motion_vertical_field_select[1][1]$ は、第2ベクトルの後方向の動きベクトルを生成する際の参照フィールドを示している。

`coded_block_pattern` は、DCT 係数を格納する複数の DCT ブロックのうち、どの DCT ブロックに、有意係数（非 0 係数）があるかを示す可変長のデータである。
`num_mv_bits` は、マクロブロック中の動きベクトルの符号量を示すデータである。
`num_coef_bits` は、マクロブロック中の DCT 係数の符号量を示すデータである。
`num_other_bits` は、マクロブロックの符号量で、動きベクトル及び DCT 係数以外の符号量を示すデータである。

次に、可変長の履歴ストリームから各データエレメントをデコードするためのシンタックスについて、図 4 7 乃至図 6 7 を参照して説明する。

この可変長の履歴ストリームは、`next_start_code()` 関数、`sequence_header()` 関数、`sequence_extension()` 関数、`extension_and_user_data(0)` 関数、`group_of_picture_header()` 関数、`extension_and_user_data(1)` 関数、`picture_header()` 関数、`picture_coding_extension()` 関数、`re_coding_stream_info()` 関数、`extension_and_user_data(2)` 関数、及び `picture_data()` 関数によって定義されたデータエレメントによって構成される。

`next_start_code()` 関数は、ビットストリーム中に存在するスタートコードを探すための関数であるので、履歴ストリームの最も先頭には、図 4 8 に示すような、過去の符号化処理において使用されたデータエレメントであって `sequence_header()` 関数によって定義されたデータエレメントが記述されている。

`sequence_header()` 関数によって定義されたデータエレメントは、`sequence_header_code`、`sequence_header_present_flag`、`horizontal_size_value`、`vertical_size_value`、`aspect_ratio_information`、`frame_rate_code`、`bit_rate_value`、`marker_bit`、`VBV_buffer_size_value`、`constrained_parameter_flag`、`load_intra_quantiser_matrix`、`intra_quantiser_matrix`、`load_non_intra_quantiser_matrix`、及び `non_intra_quantiser_matrix` 等である。

`sequence_header_code` は、シーケンス層のスタート同期コードを表すデータ

である。sequence_header_present_flag は、sequence_header 内のデータが有効か無効かを示すデータである。horizontal_size_value は、画像の水平方向の画素数の下位 12 ビットから成るデータである。vertical_size_value は、画像の縦のライン数の下位 12 ビットからなるデータである。

aspect_ratio_information は、画素のアスペクト比（縦横比）または表示画面アスペクト比を表すデータである。frame_rate_code は、画像の表示周期を表すデータである。bit_rate_value は、発生ビット量に対する制限のためのビット・レートの下位 18 ビット(400bsp 単位で切り上げる)データである。

marker_bit は、スタートコードエミュレーションを防止するために挿入されるビットデータである。VBV_buffer_size_value は、発生符号量制御用の仮想バッファ（ビデオバッファベリファイヤー）の大きさを決める値の下位 10 ビットデータである。constrained_parameter_flag は、各パラメータが制限以内であることを示すデータである。load_intra_quantiser_matrix は、イントラ MB 用量子化マトリックス・データの存在を示すデータである。

intra_quantiser_matrix は、イントラ MB 用量子化マトリックスの値を示すデータである。load_non_intra_quantiser_matrix は、非イントラ MB 用量子化マトリックス・データの存在を示すデータである。non_intra_quantiser_matrix は、非イントラ MB 用量子化マトリックスの値を表すデータである。

sequence_header() 関数によって定義されたデータエレメントの次には、図 4 9 で示すような、sequence_extension() 関数によって定義されたデータエレメントが、履歴ストリームとして記述されている。

sequence_extension() 関数によって定義されたデータエレメントとは、extension_start_code、extension_start_code_identifier、sequence_extension_present_flag、profile_and_level_indication、progressive_sequence、chroma_format、horizontal_size_extension、vertical_size_extension、bit_rate_extension、vbu_buffer_size_extension、low_delay、frame_rate_extension_n、及び frame_rate_extension_d 等のデー

タエレメントである。

`extension_start_code` は、エクステンションデータのスタート同期コードを表すデータである。`extension_start_code_identifier` は、どの拡張データが送られるかを示すデータである。`sequence_extension_present_flag` は、シーケンスエクステンション内のデータが有効であるか無効であるかを示すデータである。`profile_and_level_indication` は、ビデオデータのプロファイルとレベルを指定するためのデータである。`progressive_sequence` は、ビデオデータが順次走査であることを示すデータである。`chroma_format` は、ビデオデータの色差フォーマットを指定するためのデータである。`horizontal_size_extension` は、シーケンスヘッダの `horizontal_size_value` に加える上位 2 ビットのデータである。`vertical_size_extension` は、シーケンスヘッダの `vertical_size_value` に加える上位 2 ビットのデータである。`bit_rate_extension` は、シーケンスヘッダの `bit_rate_value` に加える上位 12 ビットのデータである。

`vbv_buffer_size_extension` は、シーケンスヘッダの `vbv_buffer_size_value` に加える上位 8 ビットのデータである。

`low_delay` は、B ピクチャを含まないことを示すデータである。

`frame_rate_extension_n` は、シーケンスヘッダの `frame_rate_code` と組み合わせてフレームレートを求めるためのデータである。`frame_rate_extension_d` は、シーケンスヘッダの `frame_rate_code` と組み合わせてフレームレートを求めるためのデータである。

`sequence_extension()` 関数によって定義されたデータエレメントの次には、図 50 に示すような `extension_and_user_data(0)` 関数によって定義されたデータエレメントが、履歴ストリームとして記述されている。

`extension_and_user_data(i)` 関数は、「i」が 1 以外のときは、`extension_data()` 関数によって定義されるデータエレメントは記述せずに、`user_data()` 関数によって定義されるデータエレメントのみを履歴ストリームとして記述する。よって、

`extension_and_user_data(0)`関数は、
`user_data()`関数によって定義されるデータエレメントのみを履歴ストリームとして記述する。

`user_data()`関数は、図51に示されたようなシンタックスに基いて、ユーザデータを履歴ストリームとして記述する。

`extension_and_user_data(0)`関数によって定義されたデータエレメントの次には、図52に示すような `group_of_picture_header()`関数によって定義されたデータエレメント、及び `extension_and_user_data(1)`関数によって定義されるデータエレメントが、履歴ストリームとして記述されている。但し、履歴ストリーム中に、GOP層のスタートコードを示す `group_start_code` が記述されている場合にのみ、

`group_of_picture_header()`関数によって定義されたデータエレメント、及び `extension_and_user_data(1)`関数によって定義されるデータエレメントが記述されている。

`group_of_picture_header()`関数によって定義されたデータエレメントは、`group_start_code`、`group_of_picture_header_present_flag`、`time_code`、`closed_gop`、及び `broken_link` から構成される。

`group_start_code` は、GOP層の開始同期コードを示すデータである。
`group_of_picture_header_present_flag` は、

`group_of_picture_header`内のデータエレメントが有効であるか無効であるかを示すデータである。`time_code` は、GOPの先頭ピクチャのシーケンスの先頭からの時間を示すタイムコードである。`closed_gop` は、GOP内の画像が他のGOPから独立再生可能なことを示すフラグデータである。`broken_link` は、編集などのためにGOP内の先頭のBピクチャが正確に再生できないことを示すフラグデータである。

`extension_and_user_data(1)`関数は、

`extension_and_user_data(0)`関数と同じように、`user_data()`関数によって定

義されるデータエレメントのみを履歴ストリームとして記述する。

もし、履歴ストリーム中に、GOP 層のスタートコードを示す `group_start_code` が存在しない場合には、これらの `group_of_picture_header()` 関数及び `extension_and_user_data(1)` 関数によって定義されるデータエレメントは、履歴ストリーム中には記述されていない。その場合には、

`extension_and_user_data(0)` 関数によって定義されたデータエレメントの次に、`picture_headr()` 関数によって定義されたデータエレメントが履歴ストリームとして記述されている。

`picture_headr()` 関数によって定義されたデータエレメントは、図 5 3 に示すように、`picture_start_code`、`temporal_reference`、`picture_coding_type`、`vbv_delay`、`full_pel_forward_vector`、`forward_f_code`、`full_pel_backward_vector`、`backward_f_code`、`extra_bit_picture`、及び `extra_information_picture` である。

具体的には、`picture_start_code` は、ピクチャ層の開始同期コードを表すデータである。`temporal_reference` は、ピクチャの表示順を示す番号で GOP の先頭でリセットされるデータである。`picture_coding_type` は、ピクチャタイプを示すデータである。`vbv_delay` は、ランダムアクセス時の仮想バッファの初期状態を示すデータである。`full_pel_forward_vector` は、順方向動きベクトルの精度が整数単位か半画素単位かを示すデータである。`forward_f_code` は、順方向動きベクトル探索範囲を表すデータである。`full_pel_backward_vector` は、逆方向動きベクトルの精度が整数単位か半画素単位かを示すデータである。

`backward_f_code` は、逆方向動きベクトル探索範囲を表すデータである。

`extra_bit_picture` は、後続する追加情報の存在を示すフラグである。この `extra_bit_picture` が「1」の場合には、次に `extra_information_picture` が存在し、`extra_bit_picture` が「0」の場合には、これに続くデータが無いことを示している。`extra_information_picture` は、規格において予約された情報である。

picture_headr()関数によって定義されたデータエレメントの次には、図54に示すようなpicture_coding_extension()関数によって定義されたデータエレメントが、履歴ストリームとして記述されている。

このpicture_coding_extension()関数によって定義されたデータエレメントとは、extension_start_code、extension_start_code_identifier、f_code[0][0]、f_code[0][1]、f_code[1][0]、f_code[1][1]、intra_dc_precision、picture_structure、top_field_first、frame_predictive_frame_dct、concealment_motion_vectors、q_scale_type、intra_vlc_format、alternate_scan、repeat_firt_field、chroma_420_type、progressive_frame、composite_display_flag、v_axis、field_sequence、sub_carrier、burst_amplitude、及びsub_carrier_phaseから構成される。

extension_start_codeは、ピクチャ層のエクステンションデータのスタートを示す開始コードである。extension_start_code_identifierは、どの拡張データが送られるかを示すコードである。f_code[0][0]は、フォワード方向の水平動きベクトル探索範囲を表すデータである。f_code[0][1]は、フォワード方向の垂直動きベクトル探索範囲を表すデータである。f_code[1][0]は、バックワード方向の水平動きベクトル探索範囲を表すデータである。f_code[1][1]は、バックワード方向の垂直動きベクトル探索範囲を表すデータである。

intra_dc_precisionは、DC係数の精度を表すデータである。

picture_structureは、フレームストラクチャかフィールドストラクチャかを示すデータである。フィールドストラクチャの場合は、上位フィールドか下位フィールドかもあわせて示すデータである。top_field_firstは、フレームストラクチャの場合、最初のフィールドが上位か下位かを示すデータである。

frame_predictive_frame_dctは、フレーム・ストラクチャの場合、フレーム・モードDCTの予測がフレーム・モードだけであることを示すデータである。

concealment_motion_vectorsは、イントラマクロブロックに伝送エラーを隠蔽するための動きベクトルがついていることを示すデータである。q_scale_type

は、線形量子化スケールを利用するか、非線形量子化スケールを利用するかを示すデータである。intra_vlc_format は、イントラマクロブロックに、別の2次元VLCを使うかどうかを示すデータである。

alternate_scan は、ジグザグスキャンを使うか、オルタネート・スキャンを使うかの選択を表すデータである。repeat_first_field は、2 : 3 プルダウンの際に使われるデータである。chroma_420_type は、信号フォーマットが4 : 2 : 0 の場合、次の progressive_frame と同じ値、そうでない場合は0を表すデータである。progressive_frame は、このピクチャが、順次走査できているかどうかを示すデータである。composite_display_flag は、ソース信号がコンポジット信号であったかどうかを示すデータである。v_axis は、ソース信号が、PAL の場合に使われるデータである。field_sequence は、ソース信号が、PAL の場合に使われるデータである。sub_carrier は、ソース信号が、PAL の場合に使われるデータである。burst_amplitude は、ソース信号が、PAL の場合に使われるデータである。sub_carrier_phase は、ソース信号が、PAL の場合に使われるデータである。

picture_coding_extension() 関数によって定義されたデータエレメントの次には、re_coding_stream_info() 関数によって定義されたデータエレメントが履歴ストリームとして記述されている。この re_coding_stream_info() 関数は、本発明の特徴的な関数であって、主に履歴情報の組み合わせを記述する場合に用いられるものであり、その詳細については、図68を参照して後述する。

re_coding_stream_info() 関数によって定義されたデータエレメントの次には、extensions_and_user_data(2) によって定義されたデータエレメントが、履歴ストリームとして記述されている。この extension_and_user_data(2) 関数は、図50に示したように、ビットストリーム中にエクステンションスタートコード (extension_start_code) が存在する場合には、extension_data() 関数によって定義されるデータエレメントが記述されている。このデータエレメントの次には、ビットストリーム中にユーザデータスタートコード (user_data_start_code)

が存在する場合には、`user_data()` 関数によって定義されるデータエレメントが記述されている。但し、ビットストリーム中にエクステンションスタートコード及びユーザデータスタートコードが存在しない場合には `extension_data()` 関数及び `user_data()` 関数によって定義されるデータエレメントはビットストリーム中には記述されていない。

`extension_data()` 関数は、図 5 5 に示すように、`extension_start_code` を示すデータエレメントと、`quant_matrix_extension()` 関数、`copyright_extension()` 関数、及び `picture_display_extension()` 関数によって定義されるデータエレメントとを、ビットストリーム中に履歴ストリームとして記述するための関数である。

`quant_matrix_extension()` 関数によって定義されるデータエレメントは、図 5 6 に示すように、`extension_start_code`、`extension_start_code_identifier`、`quant_matrix_extension_present_flag`、`load_intra_quantiser_matrix`、`intra_quantiser_matrix[64]`、`load_non_intra_quantiser_matrix`、`non_intra_quantiser_matrix[64]`、`load_chroma_intra_quantiser_matrix`、`chroma_intra_quantiser_matrix[64]`、`load_chroma_non_intra_quantiser_matrix`、及び `chroma_non_intra_quantiser_matrix[64]` である。

`extension_start_code` は、この量子化マトリックスエクステンションのスタートを示す開始コードである。`extension_start_code_identifier` は、どの拡張データが送られるかを示すコードである。

`quant_matrix_extension_present_flag` は、この量子化マトリックスエクステンション内のデータエレメントが有効か無効かを示すためのデータである。

`load_intra_quantiser_matrix` は、イントラマクロブロック用の量子化マトリックスデータの存在を示すデータである。`intra_quantiser_matrix` は、イントラマクロブロック用の量子化マトリックスの値を示すデータである。

`load_non_intra_quantiser_matrix` は、非イントラマクロブロック用の量子化

マトリックスデータの存在を示すデータである。non_intra_quantiser_matrix は、非イントラマクロブロック用の量子化マトリックスの値を表すデータである。load_chroma_intra_quantiser_matrix は、色差イントラマクロブロック用の量子化マトリックス・データの存在を示すデータである。

chroma_intra_quantiser_matrix は、色差イントラマクロブロック用の量子化マトリックスの値を示すデータである。load_chroma_non_intra_quantiser_matrix は、色差非イントラマクロブロック用の量子化マトリックス・データの存在を示すデータである。chroma_non_intra_quantiser_matrix は、色差非イントラマクロブロック用の量子化マトリックスの値を示すデータである。

copyright_extension()関数によって定義されるデータエレメントは、図57に示すように、

extension_start_code、extension_start_code_identifier、copyright_extension_present_flag、copyright_flag、copyright_identifier、original_or_copy、copyright_number_1、copyright_number_2、及びcopyright_number_3 から構成される。

extension_start_code は、コピーライトエクステンションのスタートを示す開始コードである。extension_start_code_identifier どのエクステンションデータが送られるかを示すコードである。copyright_extension_present_flag は、このコピーライトエクステンション内のデータエレメントが有効か無効かを示すためのデータである。

copyright_flag は、次のコピーライトエクステンション又はシーケンスエンドまで、符号化されたビデオデータに対してコピー権が与えられているか否かを示す。copyright_identifier は、ISO/IEC JTC/SC29 によって指定されたコピー権の登録機関を識別するためのデータである。original_or_copy は、ビットストリーム中のデータが、オリジナルデータであるかコピーデータであるかを示すデータである。copyright_number_1 は、コピーライトナンバーのビット44から63を表わすデータである。copyright_number_2 は、コピーライトナンバー

のビット 22 から 43 を表わすデータである。copyright_number_3 は、コピーライトナンバーのビット 0 から 21 を表わすデータである。

picture_display_extension() 関数によって定義されるデータエレメントは、図 58 に示すように、extension_start_code_identifier、frame_center_horizontal_offset、frame_center_vertical_offset 等である。

extension_start_code_identifier は、どの拡張データが送られるかを示すコードである。frame_center_horizontal_offset は、表示エリアの水平方向のオフセットを示すデータであって、number_of_frame_center_offsets によって定義される数のオフセット値を定義することができる。

frame_center_vertical_offset は、表示エリアを垂直方向のオフセットを示すデータであって、

number_of_frame_center_offsets によって定義される数のオフセット値を定義することができる。

再び図 47 に戻って、extension_and_user_data(2) 関数によって定義されるデータエレメントの次には、picture_data() 関数によって定義されるデータエレメントが、履歴ストリームとして記述されている。但し、この picture_data() 関数は、red_bw_flag が 1 ではないか、または、red_bw_indicator が 2 以下である場合に存在する。この red_bw_flag と red_bw_indicator は、re_coding_stream_info() 関数に記述されており、これらについては、図 68 と図 69 を参照して後述する。

picture_data() 関数によって定義されるデータエレメントは、図 59 に示すように、slice() 関数によって定義されるデータエレメントである。この slice() 関数によって定義されるデータエレメントはビットストリーム中に少なくとも 1 個記述されている。

slice() 関数は、図 60 に示されるように、slice_start_code、slice_quantiser_scale_code、intra_slice_flag、intra_slice、reserved_bits、extra_bit_slice、extra_information_slice、及び extra_bit_slice 等のデ

ータエレメントと、macroblock()関数によって定義されるデータエレメントを、履歴ストリームとして記述するための関数である。

slice_start_code は、slice()関数によって定義されるデータエレメントのスタートを示すスタートコードである。slice_quantiser_scale_code は、このスライス層に存在するマクロブロックに対して設定された量子化ステップサイズを示すデータである。しかし、各マクロブロック毎に、quantiser_scale_code が設定されている場合には、各マクロブロックに対して設定された macroblock_quantiser_scale_code のデータが優先して使用される。

intra_slice_flag は、ビットストリーム中に intra_slice 及び reserved_bits が存在するか否かを示すフラグである。intra_slice は、スライス層中にノンイントラマクロブロックが存在するか否かを示すデータである。スライス層におけるマクロブロックのいずれかがノンイントラマクロブロックである場合には、intra_slice は「0」となり、スライス層におけるマクロブロックの全てがノンイントラマクロブロックである場合には、intra_slice は「1」となる。

reserved_bits は、7ビットのデータであって「0」の値を取る。

extra_bit_slice は、履歴ストリームとして追加の情報が存在することを示すフラグであって、次に extra_information_slice が存在する場合には「1」に設定される。追加の情報が存在しない場合には「0」に設定される。

これらのデータエレメントの次には、macroblock()関数によって定義されたデータエレメントが、履歴ストリームとして記述されている。

macroblock()関数は、図61に示すように、macroblock_escape、macroblock_address_increment、及び macroblock_quantiser_scale_code、及び marker_bit 等のデータエレメントと、macroblock_modes()関数、motion_vectors(s)関数、及び code_block_pattern()関数によって定義されたデータエレメントを記述するための関数である。

macroblock_escape は、参照マクロブロックと前のマクロブロックとの水平方向の差が34以上であるか否かを示す固定ビット列である。参照マクロブロック

と前のマクロブロックとの水平方向の差が3 4以上の場合には、
macroblock_address_increment の値に3 3をプラスする。
macroblock_address_increment は、参照マクロブロックと前のマクロブロックとの水平方向の差を示すデータである。もし、この
macroblock_address_increment の前に macroblock_escape が1 存在するのであれば、この macroblock_address_increment の値に3 3をプラスした値が、実際の参照マクロブロックと前のマクロブロックとの水平方向の差分を示すデータとなる。

macroblock_quantiser_scale_code は、各マクロブロック毎に設定された量子化ステップサイズであり、macroblock_quant が” 1 ” のときだけ存在する。各スライス層には、スライス層の量子化ステップサイズを示す
slice_quantiser_scale_code が設定されているが、参照マクロブロックに対して macroblock_quantiser_scale_code が設定されている場合には、この量子化ステップサイズを選択する。

macroblock_address_increment の次には、macroblock_modes() 関数によって定義されるデータエレメントが記述されている。macroblock_modes() 関数は、図6 2に示すように、macroblock_type、frame_motion_type、field_motion_type、dct_type 等のデータエレメントを、履歴ストリームとして記述するための関数である。

macroblock_type は、マクロブロックの符号化タイプを示すデータである。その詳細は、図6 5乃至図6 7を参照して後述する。

もし、macroblock_motion_forward 又は macroblock_motion_backward が「1」であり、ピクチャ構造がフレームであり、さらに frame_pred_frame_dct が「0」である場合には、macroblock_type を表わすデータエレメントの次に frame_motion_type を表わすデータエレメントが記述されている。尚、この frame_pred_frame_dct は、

frame_motion_type がビットストリーム中に存在するか否かを示すフラグであ

る。

frame_motion_type は、フレームのマクロブロックの予測タイプを示す2ビットのコードである。予測ベクトルが2個でフィールドベースの予測タイプであれば「00」であって、予測ベクトルが1個でフィールドベースの予測タイプであれば「01」であって、予測ベクトルが1個でフレームベースの予測タイプであれば「10」であって、予測ベクトルが1個でディアルプライムの予測タイプであれば「11」である。

frame_motion_type を記述する条件が満足されない場合には、macroblock_type を表わすデータエレメントの次に field_motion_type を表わすデータエレメントが記述されている。

field_motion_type は、フィールドのマクロブロックの動き予測を示す2ビットのコードである。予測ベクトルが1個でフィールドベースの予測タイプであれば「01」であって、予測ベクトルが2個で18×8マクロブロックベースの予測タイプであれば「10」であって、予測ベクトルが1個でディアルプライムの予測タイプであれば「11」である。

もし、ピクチャ構造がフレームで、

frame_pred_frame_dct が frame_motion_type がビットストリーム中に存在することを示し、且つ、frame_pred_frame_dct が dct_type がビットストリーム中に存在することを示している場合には、macroblock_type を表わすデータエレメントの次に dct_type を表わすデータエレメントが記述されている。尚、dct_type は、DCT がフレーム DCT モードか、フィールド DCT モードかを示すデータである。

再び図61に戻って、もし、参照マクロブロックが前方予測マクロブロックであるか、又は参照マクロブロックがイントラマクロブロックであって且つコンシール処理のマクロブロックのいずれかの場合には、motion_vectors(0)関数によって定義されるデータエレメントが記述される。また、参照マクロブロックが後方予測マクロブロックである場合には、motion_vectors(1)関数によって定義さ

れるデータエレメントが記述される。尚、

`motion_vectors(0)`関数は、第1番目の動きベクトルに関するデータエレメントを記述するための関数であって、`motion_vectors(1)`関数は、第2番目の動きベクトルに関するデータエレメントを記述するための関数である。

`motion_vectors(s)`関数は、図63に示されるように、動きベクトルに関するデータエレメントを記述するための関数である。

もし、動きベクトルが1個でディアルプライム予測モードを使用していない場合には、`motion_vertical_field_select[0][s]`と`motion_vector(0, s)`によって定義されるデータエレメントが記述される。

この`motion_vertical_field_select[r][s]`は、第1番目の動きベクトル（前方又は後方のどちらのベクトルであっても良い）が、ボトムフィールドを参照して作られたベクトルであるかトップフィールドを参照して作られたベクトルであるかを示すフラグである。この指標“r”は、第1番目のベクトル又は第2番目のベクトルのいずれのベクトルであるかを示す指標であって、“s”は、予測方向が前方又は後方予測のいずれであるかを示す指標である。

`motion_vector(r, s)`関数は、図64に示されるように、`motion_code[r][s][t]`に関するデータ列と、`motion_residual[r][s][t]`に関するデータ列と、`dmvector[t]`を表わすデータとを記述するための関数である。

`motion_code[r][s][t]`は、動きベクトルの大きさを-16～+16の範囲で表わす可変長のデータである。`motion_residual[r][s][t]`は、動きベクトルの残差を表わす可変長のデータである。よって、この`motion_code[r][s][t]`と`motion_residual[r][s][t]`との値によって詳細な動きベクトルを記述することができる。`dmvector[t]`は、ディアルプライム予測モードのときに、一方のフィールド（例えばボトムフィールドに対してトップフィールドを一方のフィールドとする）における動きベクトルを生成するために、時間距離に応じて既存の動きベクトルがスケールされると共に、トップフィールドとボトムフィールドとのライン間の垂直方向のずれを反映させるために垂直方向に対して補正を行うデータ

である。この指標“r”は、第1番めのベクトル又は第2番めのベクトルのいずれのベクトルであるかを示す指標であって、“s”は、予測方向が前方又は後方予測のいずれであるかを示す指標である。“s”は、動きベクトルが垂直方向の成分であるか水平方向の成分であるかを示すデータである。

図64に示されmotion_vector(r,s)関数によって、まず、水平方向のmotion_coder[r][s][0]を表わすデータ列が、履歴ストリームとして記述される。motion_residual[0][s][t]及びmotion_residual[1][s][t]の双方のビット数は、f_code[s][t]で示されるので、

f_code[s][t]が1でない場合には、

motion_residual[r][s][t]がビットストリーム中に存在することを示すことになる。水平方向成分のmotion_residual[r][s][0]が「1」でなくて、水平方向成分のmotion_code[r][s][0]が「0」でないということは、ビットストリーム中にmotion_residual[r][s][0]を表わすデータエレメントが存在し、動きベクトルの水平方向成分が存在するということを意味しているので、その場合には、水平方向成分のmotion_residual[r][s][0]を表わすデータエレメントが記述されている。

続いて、垂直方向のmotion_coder[r][s][1]を表わすデータ列が、履歴ストリームとして記述される。同じようにmotion_residual[0][s][t]及びmotion_residual[1][s][t]の双方のビット数は、f_code[s][t]で示されるので、

f_code[s][t]が1でない場合には、

motion_residual[r][s][t]がビットストリーム中に存在することを表わすことになる。motion_residual[r][s][1]が「1」でなくて、motion_code[r][s][1]が「0」でないということは、ビットストリーム中にmotion_residual[r][s][1]を表わすデータエレメントが存在し、動きベクトルの垂直方向成分が存在するということを意味しているので、その場合には、垂直方向成分のmotion_residual[r][s][1]を表わすデータエレメントが記述されている。

次に、図 6 5 乃至図 6 7 を参照して、macroblock_type について説明する。
macroblock_type は、macroblock_quant、dct_type_flag、
macroblock_motion_forward、及び macroblock_motion_backward などのフラグから生成された可変長データである。 macroblock_quant は、マクロブロックに対して量子化ステップサイズを設定するための macroblock_quantiser_scale_code が設定されているか否かを示すフラグあつて、ビットストリーム中に macroblock_quantiser_scale_code が存在する場合には、
macroblock_quant は「1」の値を取る。

dct_type_flag は、参照マクロブロックがフレーム DCT 又はフィールド DCT で符号化されているかを示す dct_type が存在するか否かを示すためのフラグ（言い換えると DCT されているか否かを示すフラグ）であつて、ビットストリーム中に dct_type が存在する場合には、この dct_type_flag は「1」の値を取る。
macroblock_motion_forward は、参照マクロブロックが前方予測されているか否かを示すフラグであつて、前方予測されている場合には「1」の値を取る。
macroblock_motion_backward は、参照マクロブロックが後方予測されているか否かを示すフラグであつて、後方予測されている場合には「1」の値を取る。

なお、可変長フォーマットにおいては、伝送するビットレートを減少させるために、履歴情報を削減することができる。

すなわち、macroblock_type と motion_vectors() は転送するが、quantiser_scale_code を転送しない場合には、slice_quantiser_scale_code を "00000" とすることで、ビットレートを減少させることができる。

また、macroblock_type のみ転送し、motion_vectors()、quantiser_scale_code、および dct_type を転送しない場合には、macroblock_type として、"not coded" を使用することで、ビットレートを減少することができる。

さらにまた、picture_coding_type のみ転送し、slice() 以下の情報は全て転送しない場合には、slice_start_code を持たない picture_data() を使用するこ

とで、ビットレートを減少させることができる。

以上においては、user_data 内の 23 ビットの連続する "0" が出ないようにする場合に、22 ビット毎に "1" を挿入するようにしたが、22 ビット毎でなくてもよい。また、連続する "0" の個数を数えて "1" を挿入するのではなく、Byte_align を調べて挿入するようにすることも可能である。

さらに、MPEG においては、23 ビットの連続する "0" の発生を禁止しているが、実際には、バイトの先頭から 23 ビット連続する場合だけが問題とされ、バイトの先頭ではなく、途中から 0 が 23 ビット連続する場合は、問題とされない。従って、例えば 24 ビット毎に、LSB 以外の位置に "1" を挿入するようにしてもよい。

また、以上においては、履歴情報を、video elementary stream に近い形式にしたが、packetized elementary stream や transport stream に近い形式にしてもよい。また、Elementary Stream の user_data の場所を、picture_data の前としたが、他の場所にすることもできる。

図 15 において説明したトランスコード 101 は、複数世代の符号化パラメータを履歴情報として後段のプロセスに提供するようにしたが、実際には、前述した履歴情報の全てが必要となるわけではない。例えば、比較的記憶容量に制限の無い大容量の記録媒体を備えた記録再生システムが、このトランスコードの後段に接続されている場合には、符号化パラメータの中に前述した全ての履歴情報が記述されていても問題はないが、比較的記憶容量が小さい記録媒体を備えた記録再生装置が、このトランスコードの後段に接続されている場合には、符号化ストリームのデータレートを少しでも低減するために、符号化パラメータの中に前述した全ての履歴情報を記述するのではなく、必要な履歴情報を記述する方が望ましい。また、他の例として、比較的伝送容量に制限の無い大容量の伝送容量を有した伝送路が、このトランスコードの後段に接続されている場合には、符号化パラメータの中に前述した全ての履歴情報が記述されていても問題はないが、比較的伝送容量が少ない伝送路が、このトランスコードの後段に接続されている場合

には、符号化ストリームのデータレートを少しでも低減するために、符号化パラメータの中に前述した全ての履歴情報を記述するのではなく、必要な履歴情報を記述する方が望ましい。

本発明の特徴的な点は、トランスコーディング装置の後段に接続される様々なアプリケーションのそれぞれに必要な履歴情報を、アプリケーションに応じて適応的にかつ選択的に、符号化ストリーム中に記述するものである。これを実現するために、本実施例では、符号化ストリー中に、`re-coding_stream_info` という情報を記述している。

以下に、図 6 8 を参照して、`re-coding_stream_info` のシンタックス及びデータエレメントについて詳細に説明する。

図 6 8 に示すように、`re_coding_stream_info()` 関数は、
`user_data_start_code`, `re_coding_stream_info_ID`, `red_bw_flag`,
`red_bw_indicator`, `marker_bit`, `num_other_bits`, `num_mv_bits`, `num_coef_bits`
などのデータエレメントより構成される。

`user_data_start_code` は、`user_data` が開始することを表すスタートコードである。`re_coding_stream_info_ID` は、16 ビットの整数であり、`re_coding_stream_info()` 関数の識別のために用いられる。その値は、具体的には、"1001 0001 1110 1100" (0x91ec) とされる。

`red_bw_flag` は、1 ビットのフラグであり、全ての履歴情報に関するコーディングパラメータを伝送する場合には 0 とされ、履歴情報に関するコーディングパラメータが選択的に伝送する場合には 1 とされる。具体的には、図 6 9 に示すように、`red_bw_flag` が 1 である場合、このフラグに続く `red_bw_indicator` を調べることにより、5 つのデータセットのうちどのデータセットを使用して履歴情報に関するコーディングパラメータが送られているのかを決定することができる。このデータセットは、`history_stream()` として、再符号化された符号化ストリームと一緒に伝送されるコーディングパラメータの組み合わせを決定するための情報である。したがって、このデータセットに従って、符号化ストリーム中に

記述されるコーディングパラメータが選択される。

red_bw_indicator は、2ビットの整数であり、コーディングパラメータのデータセットを定義するためのインジケータである。具体的には、図69に示すように、red_bw_indicator は、データセット2からデータセット5のうち、どのデータセットを表すかを示すデータである。

よって、符号化ストリーム中に記述されている、この red_bw_flag と red_bw_indicator とを参照することによって、5つのデータセットのうちどのデータセットを使用して履歴情報に関するコーディングパラメータが送られているのかを決定することができる。

次に、図70を参照して、各データセットにおいて、履歴情報に関してどのようなコーディングパラメータが伝送されるかについて説明する。

履歴情報は、大別すると、picture 単位の情報と、macroblock 単位の情報に分けることができる。slice 単位の情報は、それに含まれる macroblock の情報を収集することで得ることができ、GOP 単位の情報は、それに含まれる picture 単位の情報を収集することで得ることができる。

picture 単位の情報は、1フレーム毎に1回伝送されるだけなので、符号化ストリーム中に挿入される履歴情報に対して占めるビットレートはそれほど大きくはない。これに対して、macroblock 単位の情報は、各 macroblock 毎に伝送されるため、例えば1フレームの走査線数が525本で、フィールドレートが60フィールド/秒のビデオシステムにおいて、1フレームの画素数を 720×480 とすると、macroblock 単位の情報は、1フレームあたり1350 $(= (720 / 16) \times (480 / 16))$ 回伝送することが必要となる。このため、履歴情報の相当の部分が macroblock 毎の情報で占められることになる。

そこで、本実施例においては、符号化ストリーム中に挿入される履歴情報としては、少なくとも picture 単位の情報は常に伝送するが、macroblock 単位の情報は、アプリケーションに応じて選択して伝送するようにすることで、伝送する情報量を抑制することができる。

図70に示したように、履歴情報として伝送される macroblock 単位の情報には、例えば num_coef_bits, num_mv_bits, num_other_bits, q_scale_code, q_scale_type, motion_type, mv_vert_field_sel[], mv[], mb_mfwd, mb_mbwd, mb_pattern, coded_block_pattern, mb_intra, slice_start, dct_type, mb_quant, skipped_mb などがある。これらは、SMPTE-327Mにおいて定義されている macroblock_rate_information の要素を用いて表現されたものである。

num_coef_bits は、macroblock の符号量のうち、DCT 係数に要した符号量を表す。num_mv_bits は、macroblock の符号量のうち、動きベクトルに要した符号量を表す。num_other_bits は、macroblock の符号量のうち、num_coef_bits 及び num_mv_bits 以外の符号量を表す。

q_scale_code は、macroblock に適用された q_scale_code を表す。

motion_type は、macroblock に適用された動きベクトルの type を表す。

mv_vert_field_sel[] は、macroblock に適用された動きベクトルの field select を表す。

mv[] は、macroblock に適用された動きベクトルを表す。mb_mfwd は、macroblock の予測モードが前方向予測であることを示すフラグである。mb_mbwd は、macroblock の予測モードが後方向予測であることを示すフラグである。mb_pattern は、macroblock の DCT 係数の非 0 のものの有無を示すフラグである。

coded_block_pattern は、macroblock の DCT 係数の非 0 のものの有無を DCT ブロック毎に示すフラグである。mb_intra は、macroblock が intra_macro かそうでないかを示すフラグである。slice_start は、macroblock が slice の先頭であるか否かを示すフラグである。dct_type は、macroblock が field_dct か flame_dct かを示すフラグである。

mb_quant は、macroblock が quantiser_scale_code を伝送するか否かを示すフラグである。skipped_mb は、macroblock が skipped macroblock であるか否かを

示すフラグである。

これらのコーディングパラメータは、常に全て必要であるわけではなく、トランスコーダの後段に接続されたアプリケーションに応じて、必要となるコーディングパラメータが変化する。例えば、num_coef_bits や slice_start といったコーディングパラメータは、再エンコードした際のビットストリームをできる限り元の形に戻したいという transparent という要求を有するアプリケーションにおいて必要となる。ここで、「transparent という要求」とは、入力ビットストリームに対して画質の劣化していない出力ビットストリームを生成することが可能になる要求である。

つまり、トランスコーディング処理によって、ビットレートの変更のみを要求しているようなアプリケーションにおいては、これらの num_coef_bits や slice_start といったコーディングパラメータは必要ではない。また、非常に伝送路の制限が厳しい場合には、各ピクチャの符号化タイプのみを必要としているようなアプリケーションも存在する。

このような状況を鑑みて、本実施例においては、履歴情報を伝送する際の、コーディングパラメータのデータセットの例として、例えば図 70 に示すようなデータセットを用意している。

図 70 において、各データセットの中のコーディングパラメータに対応する値「2」は、その情報が履歴情報として符号化ストリーム中に存在し、利用可能であることを意味し、「0」は、その情報が符号化ストリーム中に存在しないことを意味する。「1」は、他の情報の存在を補助する目的のため、あるいは、シンタックス上存在するが、元のビットストリーム情報とは関係がないなど、その情報自身には意味がないことを表している。例えば、slice_start は、履歴情報を伝送する際の slice の先頭の macroblock において、「1」になるが、本来のビットストリームに対して、slice が必ずしも同一位置関係にあるわけではない場合には、履歴情報としては無意味になる。

本実施例においては、選択されたデータセットに応じて、(num_coef_bits,

num_mv_bits, num_other_bits), (q_scale_code, q_scale_type),
(motion_type, mv_vert_field_sel[], mv[][][]), (mb_mfwd, mb_mbwd),
(mb_pattern), (coded_block_pattern), (mb_intra), (slice_start),
(dct_type), (mb_quant), (skipped_mb) の各コーディングパラメータが、
選択的に符号化ストリーム中に記述される。

データセット1は、完全に transparent なビットストリームを再構成することを目的としたデータセットである。このデータセット1によれば、入力ビットストリームに対して画質劣化のほとんど無い出力ビットストリームを出力することができる精度の高いトランスコーディングが実現できる。データセット2も、完全に transparent なビットストリームを再構成することを目的としたデータセットである。データセット3は、完全に transparent なビットストリームを再構成することはできないが、視覚的にほぼ transparent なビットストリームを再構成できるようにするためのデータセットである。データセット4は、transparent という観点からはデータセット3よりも劣るが、視覚上問題がないビットストリームの再構成ができるデータセットである。データセット5は、transparent という観点からはデータセット4よりも劣るが、少ない履歴情報でビットストリームの完全ではない再構成ができるデータセットである。

これらのデータセットのうち、データセットの番号が小さいものほど、機能的には上位であるが、履歴情報を伝送するのに必要となる容量が多くなる。従って、想定するアプリケーションと履歴情報に使用できる容量を考慮することによって、伝送するデータセットが決定される。

図70に示した5つのデータセットのうち、データセット1の場合、red_bw_flagは0とされ、データセット2乃至データセット5のとき、red_bw_flagは1とされる。これに対して、red_bw_indicatorは、データセット2の場合0とされ、データセット3の場合1とされ、データセット4の場合2とされ、データセット5の場合3とされる。

従って、red_bw_indicatorは、red_bw_flagが1の場合に（データセット2乃

至データセット5の場合に) 規定される。

さらに、red_bw_flag が0である場合 (データセット1の場合)、マクロブロック毎に、marker_bit, num_other_bits, num_mv_bits, num_coef_bits が記述される。これら4つのデータエレメントは、データセット2乃至データセット5の場合 (red_bw_flag が1の場合)、符号化ストリーム中には記述されない。

データセット5の場合、picture_data()関数 (図59参照) を含めて、それ以下のシンタックス要素は伝送されない。つまり、picture_data()関数に含まれる複数の slice()関数 (図60参照) に関する符号化パラメータは全く伝送されない。よって、データセット5の場合、選択された履歴情報は、picture_type などの picture 単位のコーディングパラメータのみの伝送を意図したものとなる。

データセット1乃至データセット4の場合、picture_data()関数に含まれる複数の slice()関数に関する符号化パラメータが存在する。しかしながら、この slice()関数によって決定される slice のアドレス情報と、元のビットストリームの slice のアドレス情報は、選択されたデータセットに依存する。データセット1またはデータセット2の場合、履歴情報の元となったビットストリームの slice のアドレス情報と、slice()関数によって決定される slice のアドレス情報とは、同一である必要がある。

macroblock()関数のシンタックス要素 (図61参照) は、選択されたデータセットに依存する。macroblock_escape, macroblock_address_increment, macroblock_modes()関数は、常に符号化ストリーム中に存在する。しかしながら、macroblock_escape と macroblock_address_increment の情報としての有効性は、選択されたデータセットによって決定される。履歴情報に関するコーディングパラメータのデータセットが、データセット1またはデータセット2の場合、元のビットストリームの skipped_mb 情報と同じものが伝送される必要がある。

データセット4の場合、motion_vectors()関数は符号化ストリーム中には存在しない。データセット1乃至データセット3の場合、macroblock_modes()関数の macroblock_type によって、motion_vectors()関数の符号化ストリーム中におけ

る存在が決定される。データセット3またはデータセット4の場合には、`coded_block_pattern()`関数は符号化ストリーム中には存在しない。データセット1とデータセット2の場合、`macroblock_modes()`関数の`macroblock_type`によって、`coded_block_pattern()`関数の符号化ストリーム中における存在が決定される。

`macroblock_modes()`関数のシンタックス要素(図62参照)は、選択されたデータセットに依存する。`macroblock_type`は、常に存在する。データセット4の場合、`flame_motion_type`, `field_motion_type`, `dct_type`は符号化ストリーム中には存在しない。

`macroblock_type`より得られるパラメータの情報としての有効性は、選択されたデータセットによって決定される。

データセット1またはデータセット2の場合、`macroblock_quant`は、元のビットストリームの`macroblock_quant`と同じである必要がある。データセット3またはデータセット4の場合、`macroblock_quant`は、`macroblock()`関数内の`quantiser_scale_code`の存在を表し、元のビットストリームと同じである必要はない。

データセット1乃至データセット3の場合、`macroblock_motion_forward`と`macroblock_motion_backward`は、元のビットストリームと同一である必要がある。データセットがデータセット4またはデータセット5である場合、その必要はない。

データセット1またはデータセット2の場合、`macroblock_pattern`は、元のビットストリームと同一である必要がある。データセット3の場合、`macroblock_pattern`は、`dct_type`の存在を示すのに用いられる。データセット4の場合、データセット1乃至データセット3における場合のような関係は成立しない。

履歴情報のコーディングパラメータのデータセットがデータセット1乃至データセット3の場合、`macroblock_intra`は、元のビットストリームと同一である

必要がある。データセット4の場合には、その限りでない。

次に、データセットに関する情報を含んでいる符号化ストリームに対するトランスコード101の処理及び、設定されたデータセットに基づいて、符号化ストリームを生成するトランスコード101の処理について、図15に示したトランスコード101を参照して説明する。

この図15に示したトランスコードの例は、第3世代のエンコード処理によって生成された符号化ストリームST(3rd)を受け取り、この符号化ストリームST(3rd)のGOP構造又は／及びビットレートを変換した新たな符号化ストリームST(4th)を生成するための例である。

まず最初に、復号装置102は、第3世代の符号化ストリームST(3rd)を符号化する際に使用した第3世代のコーディングパラメータ(3th)を、第3世代の符号化ストリームST(3rd)から抽出し、この抽出した符号化パラメータに基づいて、符号化ストリームST(3rd)をデコードし、ベースバンドビデオ信号を生成する。さらに、復号装置102は、抽出した第3世代のコーディングパラメータ(3th)を、ヒストリ情報多重化装置103に出力する。さらに、復号装置102は、第3世代の符号化ストリームST(3rd)のピクチャレイヤからuser_data()を抽出して、そのuser_data()ヒストリデコーディング装置104に供給する。

ヒストリデコーディング装置104は、復号装置102から供給されたuser_data()からhistry_stream()を抽出する。このhistry_stream()は可変長符号化されているデータエレメントからなるストリームであるので、ヒストリデコーディング装置104は、histry_stream()に対して可変長復号化処理を施す。その結果、それぞれが所定のデータ長を有したデータエレメントからなるストリームを生成することができる。次に、ヒストリデコーディング装置104は、可変長復号化したデータストリームのシンタックスをパーズング(parsing)。尚、パーズングとは、ストリームの構文を解釈することである。

このパーズング処理を行う際、ヒストリデコーディング装置104は、

history_stream()関数内の re_coding_stream_info() 関数の中に記述されている red_bw_flag と red_bw_indicator を参照する。ヒストリデコーディング装置 104 は、ストリーム中から抜き出した red_bw_flag と red_bw_indicator を参照することによって、受け取った history_stream()に対して、5つのデータセットのうち、どのデータセットが設定されているかを判断する。これによって、ヒストリデコーディング装置 104 は、red_bw_flag と red_bw_indicator によってデータセットの種類を決定することによって、どのコーディングパラメータが history_stream()に含まれているかを知ることができる。

具体的には、red_bw_flag = 0 の場合には、データセット 1 であるので、history_stream()中には、picture_data()関数として、num_coef_bits, num_mv_bits, num_other_bits, q_scale_code, q_scale_type, motion_type, mv_vert_field_sel[][], mv[][][], mb_mfwd, mb_mbwd, mb_pattern, coded_block_pattern, mb_intra, slice_start, dct_type, mb_quant, skipped_mb 等の全てのコーディングパラメータが記述されている。

red_bw_flag = 1 かつ red_bw_indicator = 0 の場合には、データセット 2 であるので、history_stream()中には、picture_data()関数として、q_scale_code, q_scale_type, motion_type, mv_vert_field_sel[][], mv[][][], mb_mfwd, mb_mbwd, mb_pattern, coded_block_pattern, mb_intra, slice_start, dct_type, mb_quant, skipped_mb 等のコーディングパラメータが記述されている。

red_bw_flag = 1 かつ red_bw_indicator = 1 の場合には、データセット 3 であるので、history_stream()中には、picture_data()関数として、q_scale_code, q_scale_type, motion_type, mv_vert_field_sel[][], mv[][][], mb_mfwd, mb_mbwd, mb_pattern, mb_intra, slice_start, dct_type, mb_quant, skipped_mb 等のコーディングパラメータが記述されている。

red_bw_flag = 1 かつ red_bw_indicator = 2 の場合には、データセット 4 であるので、history_stream()中には、picture_data()関数として、

q_scale_code, q_scale_type のコーディングパラメータが記述されている。

red_bw_flag = 1 かつ red_bw_indicator = 3 の場合には、データセット 5 であるので、histry_stream() 中には、picture_data() 関数として、どのコーディングパラメータも記述されていない。

ヒストリデコーディング装置 104 は、red_bw_flag と red_bw_indicator の情報を参照することによって、histry_stream() 中に含まれているコーディングパラメータを、histry_stream() 中から抽出する。図 15 に示したトランスコーダーの実施例においては、このヒストリデコーディング装置 104 トランスコーダーに供給された入力符号化ストリームは、第 3 世代の符号化処理によって生成された符号化ストリームであるので、から出力される履歴情報は、第 1 世代のコーディングパラメータと第 2 世代のコーディングパラメータである。

ヒストリ情報多重化装置 103 は、復号装置 102 から供給される第 3 世代の符号化パラメータ (3th) と、ヒストリデコーディング装置 104 から供給される過去のコーディングパラメータ (1st, 2nd) とを、復号装置 102 から供給されるベースバンドのビデオデータに、図 68 から図 73 に示したようなフォーマットに従って多重化する。

ヒストリ情報分離装置 105 は、ヒストリ情報多重化装置 103 より供給されたベースバンドビデオデータを受け取り、このベースバンドビデオデータから第 1、第 2 及び第 3 世代のコーディングパラメータ (1st, 2nd, 3rd) を抽出し、符号化装置 106 に供給する。

符号化装置 106 は、ヒストリ情報分離装置 105 からベースバンドビデオデータとコーディングパラメータ (1st, 2nd, 3rd) を受け取り、受け取ったコーディングパラメータに基づいてベースバンドビデオデータを再符号化する。この際、符号化装置 106 は、過去の符号化処理において生成されたコーディングパラメータ (1st, 2nd, 3rd) と、供給されたベースバンドビデオデータから新たに生成されたコーディングパラメータとから、符号化処理に最適なコーディングパラメータを選択する。符号化装置 106 は、供給されたベー

スバンドビデオデータから新たに生成されたコーディングパラメータを使用して符号化処理を行う場合には、符号化装置 106 は、前述した「通常符号化処理」を行い、符号化装置 106 は、過去のコーディングパラメータ (1st、2nd、3rd) のうちからいずれかのコーディングパラメータを再利用する場合には、前述した「パラメータ再利用符号化処理」を行う。

ネットワーク上に設けられたコンピュータ 100 は、復号化装置 102 のデコーディング処理及び符号化装置 106 のエンコーディング処理をコントロールするための装置である。例えば、符号化装置 106 から出力される符号化ストリームを伝送するための伝送路の容量を検出し、その伝送容量に応じて適切なデータセットを、前述した 5 つのデータセットの中から選択する。また、符号化装置 106 の出力段に接続されている装置の記憶容量を検出し、その記憶容量に応じて適切なデータセットを、前述した 5 つのデータセットの中から選択する。

符号化装置 106 は、コンピュータ 100 からデータセットを示す情報を受け取り、それに基づいて red_bw_flag 及び red_bw_indicator を生成する。コンピュータ 100 から与えられた情報が、データセット 1 の場合には red_bw_flag = 0 となり、データセット 2 の場合には、red_bw_flag = 1 かつ red_bw_indicator = 0 となり、データセット 3 の場合には、red_bw_flag = 1 かつ red_bw_indicator = 1 となり、データセット 4 の場合には、red_bw_flag = 1 かつ red_bw_indicator = 2 となり、データセット 5 の場合には、red_bw_flag = 1 かつ red_bw_indicator = 3 となる。

符号化装置 106 は、決定された red_bw_flag の値及び red_bw_indicator の値に応じて、histry_stream() として、符号化ストリーム中に記述するコーディングパラメータを選択し、選択したコーディングパラメータを histry_stream() として符号化ストリーム中に記述するとともに、red_bw_flag 及び red_bw_indicator を re_coding_stream_info() として符号化ストリーム中に記述する。尚、histry_stream() として伝送されるコーディングパラメータの選択処理は、第 1 世代、第 2 世代及び第 3 世代の過去のコーディングパラメータに対し

て、夫々行われる処理である。

符号化装置 106 は、red_bw_flag = 0 の場合には、histry_stream() 中には、picture_data() 関数として、num_coef_bits, num_mv_bits, num_other_bits, q_scale_code, q_scale_type, motion_type, mv_vert_field_sel[][], mv[][], mb_mfwd, mb_mbwd, mb_pattern, coded_block_pattern, mb_intra, slice_start, dct_type, mb_quant, skipped_mb 等の全てのコーディングパラメータを符号化ストリーム中に記述する。

符号化装置 106 は、red_bw_flag = 1 かつ red_bw_indicator = 0 の場合には、histry_stream() 中には、picture_data() 関数として、q_scale_code, q_scale_type, motion_type, mv_vert_field_sel[][], mv[][], mb_mfwd, mb_mbwd, mb_pattern, coded_block_pattern, mb_intra, slice_start, dct_type, mb_quant, skipped_mb のコーディングパラメータを符号化ストリーム中に記述する。

符号化装置 106 は、red_bw_flag = 1 かつ red_bw_indicator = 1 の場合には、histry_stream() 中には、picture_data() 関数として、q_scale_code, q_scale_type, motion_type, mv_vert_field_sel[][], mv[][], mb_mfwd, mb_mbwd, mb_pattern, mb_intra, slice_start, dct_type, mb_quant, skipped_mb のコーディングパラメータを符号化ストリーム中に記述する。

符号化装置 106 は、red_bw_flag = 1 かつ red_bw_indicator = 2 の場合には、histry_stream() 中には、picture_data() 関数として、q_scale_code, q_scale_type のコーディングパラメータを符号化ストリーム中に記述する。

符号化装置 106 は、red_bw_flag = 1 かつ red_bw_indicator = 3 の場合には、histry_stream() 中には、picture_data() 関数として、どのコーディングパラメータも符号化ストリーム中に記述しない。

すなわち、符号化装置 106 は、供給された過去のコーディングパラメータの全てを histry_stream() として伝送するのではなく、コンピュータ 100 から指定されたデータセットを示す情報に基づいて、histry_stream() として伝送され

るコーディングパラメータを選択する。従って、符号化装置105は、受け取ったデータセットを示す情報に基づいて、様々なデータ容量のhistry_stream()を含んだ符号化ストリームを生成することができる。また、符号化装置105は、符号化装置105の後段に接続される伝送メディアの伝送容量、記録メディアの記憶容量又はアプリケーションに見合った適切なデータ量からなる

histry_stream()を含んだ符号化ストリームを生成することができる。また、本実施例のトランスコードによれば、このようにhistry_stream()として伝送する符号化パラメータを、符号化装置の後段に接続されるアプリケーションに応じて、選択するようにしているので、アプリケーションに応じた履歴を最適なデータ量で伝送するようにすることができる。

次に、図71から図74を参照して、これらの履歴情報を、ヒストリ情報多重化装置103が出力するベースバンドビデオ信号に多重して伝送するフォーマットについて説明する。

図71は、ベースバンドビデオ信号に多重して伝送するフォーマット

「Re_Coding information Bus macroblock format」を示す図である。このマクロブロックは、 16×16 ($=256$) ビットで構成される。そして、そのうちの図71において上から3行目と4行目に示す32ビットが、picrate_elementとされる。このpicrate_elementには、図72乃至図74に示すPicture rate elementsが記述される。図72の上から2行目に1ビットのred_bw_flagが規定されており、また、3行目に3ビットのred_bw_indicatorが規定されている。即ち、これらのフラグred_bw_flag, red_bw_indicatorは、図71のpicrate_elementとして伝送される。

図71のその他のデータについて説明すると、SRIB_sync_codeは、このフォーマットのマクロブロックの最初の行が左詰めにアライメントされていることを表すコードであり、具体的には、“11111”に設定される。fr_fl_SRIBは、picture_structureがフレームピクチャ構造の場合（その値が“11”である場合）、1に設定され、Re_Coding Information Bus macroblockが1.6ライン

を超えて伝送されることを表し、picture_structure がフレーム構造ではない場合、0 に設定され、Re_Coding Information Bus が 16 ラインを超えて伝送されることを意味する。この機構により、Re_Coding Information Bus が、空間的かつ時間的にデコードされたビデオフレームまたはフィールドの対応する画素にロックされる。

SRIB_top_field_first は、元のビットストリームに保持されている top_field_first と同じ値に設定され、関連するビデオの Re_Coding Information Bus の時間的アライメントを repeat_first_field とともに表している。SRIB_repeat_first_field は、元のビットストリームに保持されている repeat_first_field と同じ値に設定される。first field の Re_Coding Information Bus の内容は、このフラグに示されるように繰り返される必要がある。

422_420_chroma は、元のビットストリームが 4 : 2 : 2 または 4 : 2 : 0 のいずれであるかを表す。その値の 0 は、ビットストリームが 4 : 2 : 0 であり、色差信号のアップサンプリングが、4 : 2 : 2 のビデオが出力されるように行われたことを表す。その値の 1 は、色差信号のフィルタリング処理が実行されていないことを表す。

rolling_SRIB_mb_ref は、16 ビットのモジュロ 65521 を表し、この値は、毎マクロブロック毎にインクリメントされる。この値は、フレームピクチャ構造のフレームに渡って連続している必要がある。さもなくば、この値は、フィールドに渡って連続している必要がある。この値は、0 から 65520 の間の所定の値に初期化される。これにより、レコーダのシステムに、ユニークな Re_Coding Information Bus の識別子を組み込むことが許容される。

Re_Coding Information Bus macroblock のその他のデータの意味は、上述した通りであるので、ここでは省略する。

図 75 に示すように、図 71 の 256 ビットの Re_Coding Information Bus のデータは、1 ビットずつ、色差データの LSB である Cb[0][0], Cr[0][0],

Cb[1][0], Cr[1][0]に配置される。図75に示すフォーマットにより、4ビットのデータを送ることができるので、図71の256ビットのデータは、図75のフォーマットを64 ($=256/4$) 個送ることで伝送することができる。

本発明のトランスコードによれば、過去の符号化処理において生成された符号化パラメータを、現在の符号化処理において再利用するようにしているので、復号処理及び符号化処理を繰り返したとしても画質劣化が発生しない。つまり、復号処理及び符号化処理の繰り返しによる画質劣化の蓄積を低減することができる。

図76と図77は、本発明のトランスコードをビデオテープレコードに適用した場合の構成例を表している。図76は、ビデオテープレコード601の記録系の構成例を表しており、図77は、ビデオテープレコード601の再生系の構成例を表している。

図76のビデオテープレコード601は、トランスコード101R、チャンネルエンコーディング装置602、記録ヘッド603により構成されている。トランスコード101Rの構成は、図37に示したトランスコードと基本的に同様とされている。この構成例においては、トランスコード101Rにおいては、Long GOPのビットストリームSTが、Short GOPのビットストリームSTに変換される。

トランスコード101Rの符号化装置106より出力された、第4世代の符号化ストリームSTが、チャンネルエンコーディング装置602に供給される。上述したように、この第4世代の符号化ストリームSTのピクチャ層のユーザデータエリアには、第1世代乃至第3世代の符号化パラメータを含むユーザデータ user_data が記録されている。

チャンネルエンコーディング装置602は、入力された第4世代の符号化ストリームに、誤り訂正のためのパリティ符号を付けた後、例えば、NRZI 変調方式でチャンネルエンコードし、記録ヘッド603に供給する。記録ヘッド603は入力された符号化ストリームを磁気テープ604に記録する。

図 77 に示すように、再生系においては、磁気テープ 604 から、再生ヘッド 611 により信号が生成され、チャンネルデコーディング装置 612 に供給される。チャンネルデコーディング装置 612 は、再生ヘッド 611 から供給された信号をチャンネルデコードし、パリティを用いて、誤り訂正する。

チャンネルデコーディング装置 612 により出力された第 4 世代の符号化ストリーム ST は、トランスコーダ 101P に入力される。トランスコーダ 101P の基本的な構成は、図 37 に示したトランスコーダと同様の構成とされている。

トランスコーダ 101P の復号装置 102 は、第 4 世代の符号化ストリームから、第 1 世代乃至第 3 世代の符号化パラメータを含むユーザデータ user_data を抽出し、ヒストリデコーディング装置 104 と、符号化装置 106 に供給する。ヒストリデコーディング装置 104 は、入力されたユーザデータ user_data を復号し、得られた第 1 世代乃至第 3 の符号化パラメータを符号化装置 106 に供給する。

復号装置 102 はまた、第 4 世代の符号化ストリーム ST を復号し、ベースバンドビデオ信号と、第 4 世代の符号化パラメータを出力する。ベースバンドビデオ信号は、符号化装置 106 に供給され、第 4 世代の符号化パラメータは、符号化装置 106 と、ヒストリエンコーディング装置 107 に供給される。

ヒストリエンコーディング装置 107 は、入力された第 4 世代の符号化パラメータを、ユーザデータ user_data に変換し、符号化装置 106 に供給する。

上述したように、符号化装置 106 のコントローラ 70 は、オペレータによって指定された GOP 構造から決定された各ピクチャのピクチャタイプと、ヒストリ情報（ユーザデータ user_data）に含まれるピクチャタイプが一致するか否かを判断する。そしてその判断結果に対応して、上述した、「通常符号化処理」、または「パラメータ再利用符号化処理」を実行する。この処理により符号化装置 106 から、Short GOP から Long GOP に変換された、第 4 世代の符号化ストリーム ST が出力される。この符号化ストリーム ST のユーザデータ user_data には、第 1 世代乃至第 4 世代の符号化パラメータがヒストリ情報として記録されている。

。図76と図77に示したビデオテープレコーダ601においては、ヒストリ情報をピクチャレイヤのuser_dataに記録するようにしたが、ヒストリ情報は、磁気テープ604のビデオデータとは異なる領域に記録することも可能である。図78と図79は、この場合のビデオテープレコーダ601の構成例を表している。図78は、ビデオテープレコーダ601の記録系の構成例を表しており、図79は、再生系の構成例を表している。

図78に示すように、このビデオテープレコーダ601においては、そのトランスコーダ101Rの復号装置102より出力されたユーザデータuser_dataが、ヒストリデコーディング装置104に入力され、そこで過去の符号化パラメータ（この例の場合、第1世代と第2世代の符号化パラメータ）が復号され、符号化装置106に供給されている。また、この例においては、磁気テープ604にユーザデータuser_dataとして、ヒストリ情報を記録する必要がないので、図15に示したヒストリエンコーディング装置107のうち、ヒストリVLC211のみが採用されている。そしてこのヒストリVLC211に、復号装置102が出力した符号化パラメータ（この例の場合、第3世代の符号化パラメータ）と、ヒストリデコーディング装置104がユーザデータuser_dataから復号、出力した符号化パラメータ（この例の場合、第1世代と第2世代の符号化パラメータ）が供給されている。ヒストリVLC211は、この第1世代乃至第3世代の符号化パラメータを可変長符号化し、図40乃至図46、または図47に示したhistory_streamを生成し、マルチプレクサ621に供給する。

マルチプレクサ621には、また、符号化装置106より出力された第4世代の符号化ストリームSTが入力されている。マルチプレクサ621は、符号化装置106より供給された符号化ストリーム（ビットストリーム）をヒストリVLC211より供給されたヒストリよりも安全な領域に多重化する。

例えば、図80に示すように、磁気テープ604において、符号化装置106より出力されたビデオストリームは、シンクコードに近い位置に記録され、ヒス

トリ VLC 2 1 1 より出力された history_stream は、シンクコードからビデオストリームより、より離れた位置に記録される。特殊再生時などにおいて、ビデオストリームを検索するとき、最初にシンクコードが検出され、そのシンクコードを基準として、それに続くビデオストリームが検索される。従って、シンクコードに近い位置にビデオストリームを配置した方が、高速再生時などにおいても、より確実にビデオデータを再生することが可能になる。history_stream は、高速再生時などにおいて、必要とされる情報ではない。そこで、この history_stream は、シンクコードからより離れた位置に配置しても、それほど支障は生じない。

マルチプレクサ 6 2 1 により多重化された信号は、チャンネルエンコーディング装置 6 0 2 に入力され、チャンネルエンコードされた後、記録ヘッド 6 0 3 により、磁気テープ 6 0 4 に記録される。

このように、この例においては、history_stream が、ビデオデータとは異なる位置に多重化されるため、仮にそこにスタートコードが現れたとしても、ビデオデータとは充分区別することが可能である。そこで、この例においては、マーカビットを挿入し、history_stream を converted_history_stream とする必要はない。

また、符号化パラメータを history_stream のフォーマットにしないで、そのままマルチプレクサ 6 2 1 に供給し、多重化させることも可能データあるが、そのようにすると、圧縮されていないので、符号化パラメータのデータ量が多くなり、磁気テープ 6 0 4 の利用効率が低下する。そこで、ヒストリ VLC 2 1 1 により圧縮し、history_stream のフォーマットにして、多重化するようにするのが好ましい。

図 7 9 に示すように、ビデオテープレコーダ 6 0 1 の再生系においては、磁気テープ 6 0 4 から、再生ヘッド 6 1 1 により再生された信号が、チャンネルデコーディング装置 6 1 2 でチャンネルデコードされる。デマルチプレクサ 6 3 1 は、チャンネルデコーディング装置 6 1 2 でチャンネルデコードされる。デマルチ

プレクサ631は、チャンネルデコーディング装置612から供給された第4世代の符号化ストリームSTを、ビデオストリームと、history_streamとに分離し、ビデオストリームを復号装置102に供給し、history_streamをヒストリVLD203に供給する。

すなわちこの例においては、図15に示したヒストリデコーディング装置104のうち、ヒストリVLD203のみが採用される。

ヒストリVLD203は、history_streamを可変長復号処理し、得られた第1世代乃至第3世代の符号化パラメータを符号化装置106に出力する。

また、デマルチプレクサ631より出力されたhistory_streamは、コンバータ212'入力される。このコンバータ212'と、後段のユーザデータフォーマッタ213'は、ヒストリエンコーディング装置107に内蔵されているコンバータ212、およびユーザデータフォーマッタ213（図15参照）とは、別個のものであるが、それらと同一の機能を果たすものである。

すなわちコンバータ212'は、デマルチプレクサ631より入力されたhistory_streamにマーカービットを付加して、converted_history_streamを生成し、ユーザデータフォーマッタ213'に出力する。ユーザデータフォーマッタ213'は、入力されたconverted_history_streamをuser_dataに変換し、符号化装置106に出力する。このuser_dataには、第1世代乃至第3世代の符号化パラメータが含まれていることになる。

復号装置102は、デマルチプレクサ631から入力されたビデオストリームを復号し、ベースバンドビデオ信号を符号化装置106に出力する。また、復号装置102は、第4世代の符号化パラメータを符号化装置106に供給するとともに、ヒストリ円コーディング装置107に出力する。ヒストリエンコーディング装置107は、入力された第4世代の符号化パラメータからuser_dataを生成し、符号化装置106に出力する。

符号化装置106は、図77における符号化装置106と同様に、「通常符号化処理」または「パラメータ再利用符号化処理」を実行し、第5世代の符号化ス

トリーム ST を出力する。この第 5 世代の符号化ストリーム ST には、そのピクチャ層の user_data に、第 1 世代乃至第 4 世代の符号化パラメータが記録されている。

以上の説明から理解できるように、本発明のトランスコードによれば、過去の符号化処理において生成された符号化パラメータを、現在の符号化処理において生成された符号化ストリームのユーザデータエリアに記述するようにし、生成されたビットストリームは、MPEG 規格に準じた符号化ストリームであるので、既存のどのデコーダでも復号処理を行うことができる。さらには、本発明のトランスコードによれば、過去の符号化処理における符号化パラメータを伝送するために専用線のようなものを設ける必要がないので、従来のデータストリーム伝送環境をそのまま使用して、過去の符号化パラメータを伝送することができる。

本実施例のトランスコードによれば、過去の符号化処理において生成された符号化パラメータを、選択的に現在の符号化処理において生成された符号化ストリーム中に記述するようにしているので、出力されるビットストリームのビットレートを極端に上げることなく、過去の符号化パラメータを伝送することができる。

本実施例のトランスコードによれば、過去の符号化パラメータと現在の符号化パラメータの中から、現在の符号化処理に最適な符号化パラメータを選択して符号化処理を行うようにしているので、復号処理及び符号化処理を繰り返したとしても、画質劣化が蓄積されることはない。

本実施例のトランスコードによれば、過去の符号化パラメータの中から、ピクチャタイプに応じて現在の符号化処理に最適な符号化パラメータを選択して符号化処理を行うようにしているので、復号処理及び符号化処理を繰り返したとしても、画質劣化が蓄積されることはない。

また、本実施例のトランスコードによれば、過去の符号化パラメータに含まれるピクチャタイプに基づいて、過去の符号化パラメータを再利用するか否かを決定しているので、最適な符号化処理を行うことができる。

なお、上記各処理を行うコンピュータプログラムは、磁気ディスク、光ディスク、光磁気ディスク、半導体メモリなどの記録媒体に記録して提供するほか、インターネット、ATM、デジタル衛星などのネットワークを介して伝送し、ユーザの記録媒体に記録させることで提供することができる。

また、本実施例のトランスコードによれば、トランスコードによって再符号化された符号化ストリームに対して、それまでの符号化処理における符号化履歴の組み合わせを表すデータセットに関する情報を記述するようにしたので、少ない容量の伝送路を介して伝送可能な履歴情報を含むストリームを生成することが可能となる。

本実施例のトランスコードによれば、過去の符号化処理において使用された複数の符号化パラメータを、選択的に組み合わせて、その符号化履歴情報を生成し、符号化ストリームに重畳するようにしたので、再符号化に伴う画像の劣化を抑制可能なストリームを、少ない容量のメディアを介して伝送することが可能となる。

本実施例のトランスコードによれば、データセットを示す情報に基づいて、符号化パラメータを抽出し、抽出した符号化パラメータに基づいて、再符号化された符号化ストリームを生成するようにしたので、再符号化に伴う画像の劣化を抑制した、少ない伝送容量の伝送メディアに伝送可能なストリームを符号化することが可能となる。

また、本実施例のトランスコードによれば、検出された過去の符号化処理における符号化履歴を記録媒体に記録するようにしたので、符号化ストリームを記録媒体に記録した場合においても、画質の劣化を抑制することが可能となる。

本実施例のトランスコードによれば、記録媒体から再生された符号化ストリームに含まれる符号化履歴を検出し、再符号化された符号化ストリームと多重化して出力するようにしたので、記録媒体から再生された符号化ストリームを再度トランスコーディングする場合においても、画質の劣化を抑制することが可能となる。

産業上の利用の可能性

本発明は、コーディングシステム及び方法、符号化装置および方法、復号化装置及び方法、記録装置及び方法、ならびに再生装置及び方法に関し、特に、MPEG規格に基づいて過去に符号化処理が施されたことがある符号化ストリームに対して再符号化処理を施すことによって、異なる GOP (Group of Pictures) 構造や異なるビットレートを有した再符号化ストリームを生成するためのトランスコーダなどに利用できる。

請 求 の 範 囲

1. ソース符号化ストリームに対して再符号化処理を行うコーディングシステムにおいて、

上記ソース符号化ストリームを復号してビデオデータを生成するとともに、上記ソース符号化ストリームから過去の符号化処理により生成された過去の符号化パラメータを抽出する復号手段と、

上記ビデオデータを再符号化し、再符号化ビデオストリームを生成する符号化手段と、

上記過去の符号化パラメータを受け取り、上記過去の符号化パラメータに基づいて、上記符号化手段の再符号化処理を制御するとともに、上記過去の符号化パラメータを上記再符号化ストリーム中に選択的に記述する制御手段とを備えることを特徴とするコーディングシステム。

2. 上記制御手段は、上記過去の符号化パラメータのうち、マクロブロック単位の符号化パラメータを選択的に上記再符号化ストリーム中に記述することを特徴とする請求の範囲第1項に記載のコーディングシステム。

3. 上記制御手段は、上記アプリケーションに関係なく上記過去の符号化パラメータのうちピクチャ単位に関する全ての符号化パラメータを上記再符号化ストリーム中に記述するとともに、上記アプリケーションに基づいて、上記過去の符号化パラメータのうちマクロブロック単位に関する符号化パラメータを選択的に上記再符号化ストリーム中に記述することを特徴とする請求の範囲第1項に記載のコーディングシステム。

4. 上記制御手段は、上記選択された過去の符号化パラメータを、
`history_stream ()`として上記再符号化ストリーム中に記述すると共に、上記選

択された過去の符号化パラメータのデータセットを示す情報を、
re_coding_stream_info()として上記再符号化ストリーム中に記述することを特
徴とする請求の範囲第1項に記載のコーディングシステム。

5. 上記 re_coding_stream_info()は、上記データセットを同定するための情報
として、red_bw_flag と red_bw_indicator とを含んでいることを特徴とする請
求の範囲第2項に記載のコーディングシステム。

6. 上記データセットは、複数のデータセットから構成され、少なくとも、完全
に transparent な再符号化ストリームを生成するために必要な符号化パラメータ
を伝送するデータセットと、比較的 transparent な再符号化ストリームを生成す
るために必要な符号化パラメータを伝送するデータセットとを含んでいる
ことを特徴とする請求の範囲第2項に記載のコーディングシステム

7. 上記データセットは、複数のデータセットから構成され、少なくとも、上記
再符号化パラメータに history_stream ()として上記過去の符号化パラメータの
全てを伝送するためのデータセットと、上記再符号化パラメータに
history_stream ()として上記過去の符号化パラメータの picture_data()以下の
符号化パラメータを伝送しないデータセットを含んでいる
ことを特徴とする請求の範囲第2項に記載のコーディングシステム。

8. 上記データセットは、複数のデータセットから構成され、少なくとも、上記
再符号化パラメータに history_stream ()としてピクチャ単位及びマクロブロッ
ク単位の符号化パラメータを伝送するデータセットと、上記再符号化パラメータ
に history_stream ()としてピクチャ単位の符号化パラメータは伝送するがマク
ロブロック単位の符号化パラメータは伝送しないデータセットとを含んでいる
ことを特徴とする請求の範囲第2項に記載のコーディングシステム。

9. 上記符号化手段は、上記現在の符号化処理において、上記入力ビデオデータに含まれる参照ピクチャに割り当てられた現在のピクチャタイプで、上記参照ピクチャをエンコードし、

上記制御手段は、過去の符号化処理において割り当てられたピクチャタイプと同一のピクチャタイプに上記参照ピクチャが符号化されたか否かを判定し、その判定結果に基づいて、上記現在の符号化処理を制御する

ことを特徴する請求の範囲第1項に記載のコーディングシステム。

10. 上記制御手段は、上記判定に基づいて、上記過去の符号化パラメータから最適な符号化パラメータを選択し、上記選択された最適な符号化パラメータに基づいて、上記符号化手段の現在の符号化処理を制御する

ことを特徴とする請求の範囲第9項に記載のコーディングシステム。

11. 上記制御手段は、上記参照ピクチャを、上記過去の符号化処理において発生された過去の符号化パラメータの1つを使用して符号化する

ことを特徴とする請求の範囲第9項に記載のコーディングシステム。

12. 上記過去の符号化パラメータは、上記過去の符号化処理において生成された動きベクトル情報を含み、

上記符号化手段は、上記現在の符号化処理において、上記参照ピクチャの動きベクトル情報を検出するための動きベクトル検出手段を含む

ことを特徴とする請求の範囲第9項に記載のコーディングシステム。

13. 上記制御手段は、上記判断の結果に基づいて、上記動きベクトル検出手段の動作を制御する

ことを特徴とする請求の範囲第12項に記載のコーディングシステム。

14. 上記制御手段は、上記動きベクトル検出手段における新たな動きベクトル情報の計算の代わりに、上記過去の符号化パラメータに含まれる、上記動きベクトル情報を再使用する

ことを特徴とする請求の範囲第13項に記載のコーディングシステム。

15. 上記制御手段は、上記過去の符号化パラメータから、上記現在の符号化処理に対応する最適な符号化パラメータを選択し、上記最適な符号化パラメータに基づいて、上記符号化手段の上記現在の符号化処理を制御する

ことを特徴とする請求の範囲第9項に記載のコーディングシステム。

16. 上記符号化手段は、上記現在の符号化処理において、参照ピクチャに割り当てられた現在のピクチャタイプで上記入力ビデオデータに含まれる上記参照ピクチャを符号化し、

上記制御手段は、上記参照ピクチャが過去の符号化処理において割り当てられたピクチャタイプと同一のピクチャタイプに符号化されたか否かを判定し、その判定結果に基づいて、上記最適な符号化パラメータを選択する

ことを特徴とする請求の範囲第9項に記載のコーディングシステム。

17. 上記過去の符号化パラメータは、フレーム予測モードまたはフィールド予測モードを表す予測モード情報を含み、

上記制御手段は、上記予測モード情報に対応して、上記現在の符号化処理を制御する

ことを特徴とする請求の範囲15項に記載のコーディングシステム。

18. 上記参照ピクチャが、上記過去の符号化処理において、上記過去のピクチャタイプと同一のピクチャタイプで符号化されている場合、上記制御手段は、

新たな予測モード情報の計算に代えて、上記過去の符号化パラメータに含まれる上記予測モード情報を再使用する

ことを特徴とする請求の範囲第 15 項に記載のコーディングシステム。

19. 上記符号化パラメータは、イントラ予測、前方予測、後方予測、または双方向予測を示す予測タイプ情報を含み、

上記制御手段は、上記予測タイプ情報に基づいて、上記現在の符号化処理を制御する

ことを特徴とする請求の範囲第 15 項に記載のコーディングシステム。

20. 上記符号化パラメータは、フレーム DCT モードまたはフィールド DCT モードを表す DCT モード情報を含み、

上記制御手段は、上記 DCT モード情報に基づいて、上記現在の符号化処理を制御する

ことを特徴とする請求の範囲第 15 項に記載のコーディングシステム。

21. 上記符号化手段は、シーケンスレイヤ、GOP レイヤ、ピクチャレイヤ、スライスレイヤ、およびマクロブロックレイヤを有する MPEG 標準に従った MPEG ビットストリームを生成する生成手段を有する

ことを特徴とする請求の範囲第 9 項に記載のコーディングシステム。

22. 上記制御手段は、上記符号化手段の上記現在の符号化処理に対応する現在の符号化パラメータを生成し、

上記制御手段は、上記現在の符号化パラメータを、上記再符号化ストリームの上記ピクチャレイヤ、スライスレイヤ、およびマクロブロックレイヤに記述し、上記過去の符号化パラメータを、上記再符号化ストリームのピクチャレイヤのユーザデータエリアに選択的に記述する

ことを特徴とする請求の範囲第 2 1 項に記載のコーディングシステム。

2 3. 上記制御手段は、上記過去の符号化パラメータを `history_stream()` として上記再符号化ストリーム中に記述する

ことを特徴とする請求の範囲第 9 項に記載のコーディングシステム。

2 4. ソース符号化ストリームに対して再符号化処理 (Re-Coding Process) を施すコーディングシステムにおいて、

上記ソース符号化ストリームを復号化して復号化ビデオデータを生成するとともに、上記ソース符号化ストリーム中に含まれている過去の符号化パラメータを上記ソース符号化ストリームから抽出する復号化手段と、

上記復号化ビデオデータに対して再符号化処理を施し、再符号化ストリームを生成する符号化手段と、

上記過去の符号化パラメータから、上記符号化手段の後段に接続されるアプリケーションにおいて必要となる符号化パラメータを選択し、該選択された過去の符号化パラメータを、上記再符号化ストリーム中に記述する制御手段と

を備えたことを特徴とするコーディングシステム。

2 5. ソース符号化ストリームに対して再符号化処理 (Re-Coding Process) を施すコーディング方法において、

上記ソース符号化ストリームを復号化して復号化ビデオデータを生成するとともに、上記ソース符号化ストリーム中に含まれている過去の符号化パラメータを上記ソース符号化ストリームから抽出するステップと、

上記復号化ビデオデータに対して再符号化処理を施し、再符号化ストリームを生成するステップと、

上記過去の符号化パラメータから、上記符号化手段の後段に接続されるアプリケーションにおいて必要となる符号化パラメータを選択し、該選択された過去の

符号化パラメータを、上記再符号化ストリーム中に記述するステップと
を備えたことを特徴とするコーディング方法。

26. ソース符号化ストリームに対して再符号化処理 (Re-Coding Process)
を施すコーディングシステムにおいて、

上記ソース符号化ストリームを復号化して復号化ビデオデータを生成するとともに、上記ソース符号化ストリーム中に含まれている過去の符号化パラメータを上記ソース符号化ストリームから抽出する復号化手段と、

上記復号化ビデオデータに対して再符号化処理を施し、再符号化ストリームを生成する符号化手段と、

上記過去の符号化パラメータを選択的に上記再符号化ストリーム中に記述するとともに、上記再符号化ストリーム中に記述される過去の符号化パラメータのデータセットを示すフラグ又は／及びインジケータを、上記再符号化ストリーム中に記述する制御手段と

を備えたことを特徴とするコーディングシステム。

27. ソース符号化ストリームに対して再符号化処理 (Re-Coding Process)
を施すコーディング方法において、

上記ソース符号化ストリームを復号化して復号化ビデオデータを生成するとともに、上記ソース符号化ストリーム中に含まれている過去の符号化パラメータを上記ソース符号化ストリームから抽出する復号化ステップと、

上記復号化ビデオデータに対して再符号化処理を施し、再符号化ストリームを生成する符号化ステップと、

上記過去の符号化パラメータを選択的に上記再符号化ストリーム中に記述するとともに、上記再符号化ストリーム中に記述される過去の符号化パラメータのデータセットを示すフラグ又は／及びインジケータを、上記再符号化ストリーム中に記述する制御ステップと

を備えたことを特徴とするコーディング方法、

28. ソース符号化ストリームに対して再符号化処理 (Re-Coding Process) を施すコーディングシステムにおいて、

上記ソース符号化ストリームを復号化して復号化ビデオデータを生成するとともに、上記ソース符号化ストリーム中に `history_stream()` として伝送されてきた過去の符号化パラメータを得る復号化手段と、

上記復号化ビデオデータに対して再符号化処理を施し、再符号化ストリームを生成する符号化手段と、

上記過去の符号化パラメータに関する情報を、上記符号化ストリーム中に `history_stream()` として記述するとともに、上記再符号化ストリームに関する情報を、`re_coding_stream_info()` として上記再符号化ストリーム中に記述する制御手段と

を備えたことを特徴とするコーディングシステム。

29. ソース符号化ストリームに対して再符号化処理 (Re-Coding Process) を施すコーディング方法において、

上記ソース符号化ストリームを復号化して復号化ビデオデータを生成するとともに、上記ソース符号化ストリーム中に `history_stream()` として伝送されてきた過去の符号化パラメータを得る復号化ステップと、

上記復号化ビデオデータに対して再符号化処理を施し、再符号化ストリームを生成する符号化ステップと、

上記過去の符号化パラメータに関する情報を、上記符号化ストリーム中に `history_stream()` として記述するとともに、上記再符号化ストリームに関する情報を、`re_coding_stream_info()` として上記再符号化ストリーム中に記述する制御ステップと

を備えたことを特徴とするコーディング方法。

30. ソース符号化ストリームに対して再符号化処理 (Re-Coding Process) を施すコーディングシステムにおいて、

上記ソース符号化ストリームを復号化して復号化ビデオデータを生成するとともに、上記ソース符号化ストリーム中に `history_stream()` として伝送されてきた過去の符号化パラメータを得る復号化手段と、

上記復号化ビデオデータに対して再符号化処理を施し、再符号化ストリームを生成する符号化手段と、

上記過去の符号化パラメータを選択的に上記再符号化ストリーム中に `history_stream()` として記述するとともに、上記再符号化ストリーム中に記述される過去の符号化パラメータのデータセットに関する情報を、
`re_coding_stream_info()` として上記再符号化ストリーム中に記述する制御手段と

を備えたことを特徴とするコーディングシステム。

31. ソース符号化ストリームに対して再符号化処理 (Re-Coding Process) を施すコーディング方法において、

上記ソース符号化ストリームを復号化して復号化ビデオデータを生成するとともに、上記ソース符号化ストリーム中に `history_stream()` として伝送されてきた過去の符号化パラメータを得る復号化ステップと、

上記復号化ビデオデータに対して再符号化処理を施し、再符号化ストリームを生成する符号化ステップと、

上記過去の符号化パラメータを選択的に上記再符号化ストリーム中に `history_stream()` として記述するとともに、上記再符号化ストリーム中に記述される過去の符号化パラメータのデータセットに関する情報を、
`re_coding_stream_info()` として上記再符号化ストリーム中に記述する制御ステップと

を備えたことを特徴とするコーディング方法。

32. ビデオデータを符号化する符号化装置において、

上記ビデオデータに関して施された過去の符号化処理に関する過去の符号化パラメータを受け取る手段と、

上記ビデオデータを符号化し、再符号化ストリームを生成する符号化手段と、

上記ビデオデータに関して施された過去の符号化処理に関する過去の符号化パラメータを受け取り、上記過去の符号化パラメータを選択的に上記再符号化ストリーム中に記述するとともに、上記再符号化ストリーム中に記述される上記過去の符号化パラメータのデータセットを示す情報を、上記再符号化ストリーム中に記述する制御手段と

を備えたことを特徴とする符号化装置。

33. ビデオデータを符号化する符号化方法において、

上記ビデオデータに関して施された過去の符号化処理に関する過去の符号化パラメータを受け取るステップと、

上記ビデオデータを符号化し、再符号化ストリームを生成する符号化ステップと、

上記ビデオデータに関して施された過去の符号化処理に関する過去の符号化パラメータを受け取り、上記過去の符号化パラメータを選択的に上記再符号化ストリーム中に記述するとともに、上記再符号化ストリーム中に記述される上記過去の符号化パラメータのデータセットを示す情報を、上記再符号化ストリーム中に記述する制御ステップと

から構成される符号化方法。

34. ビデオデータを符号化する符号化装置において、

上記ビデオデータに関して施された過去の符号化処理に関する過去の符号化パ

ラメータを受け取る手段と、

上記ビデオデータを符号化し、再符号化ストリームを生成する符号化手段と、
上記過去の符号化パラメータを上記再符号化ストリーム中に
`history_stream()`として記述するとともに、上記再符号化ストリーム中に記述される上記過去の符号化パラメータに関する情報を、`re_coding_stream_info()`として上記再符号化ストリーム中に記述する制御手段と
を備えたことを特徴とする符号化装置。

35. ビデオデータを符号化する符号化方法において、

上記ビデオデータに関して施された過去の符号化処理に関する過去の符号化パラメータを受け取るステップと、

上記ビデオデータを符号化し、再符号化ストリームを生成する符号化ステップと、

上記過去の符号化パラメータを上記再符号化ストリーム中に
`history_stream()`として記述するとともに、上記再符号化ストリーム中に記述される上記過去の符号化パラメータに関する情報を、`re_coding_stream_info()`として上記再符号化ストリーム中に記述する制御ステップと
から構成される符号化方法。

36. ビデオデータを符号化する符号化装置において、

上記ビデオデータに関して施された過去の符号化処理に関する過去の符号化パラメータを受け取る手段と、

上記ビデオデータに対して再符号化処理を施し、再符号化ストリームを生成する符号化手段と、

上記符号化パラメータが `history_stream()`として上記再符号化ストリーム中に記述されるとともに、上記再符号化ストリーム中に記述される符号化パラメータのデータセットに関する情報が、`re_coding_stream_info()`として上記再符号

化ストリーム中に記述されるように、上記再符号化処理をコントロールする制御手段と

を備えたことを特徴とする符号化装置。

37. ビデオデータを符号化する符号化方法において、

上記ビデオデータに関して施された過去の符号化処理に関する過去の符号化パラメータを受け取るステップと、

上記ビデオデータに対して再符号化処理を施し、再符号化ストリームを生成する符号化ステップと、

上記符号化パラメータが `history_stream()` として上記再符号化ストリーム中に記述されるとともに、上記再符号化ストリーム中に記述される符号化パラメータのデータセットに関する情報が、`re_coding_stream_info()` として上記再符号化ストリーム中に記述されるように、上記再符号化処理をコントロールする制御ステップと

から構成される符号化方法。

38. ビデオデータを符号化する符号化装置において、

上記ビデオデータに関して施された過去の符号化処理に関する過去の符号化パラメータを受け取る手段と、

上記ビデオデータを符号化し、再符号化ストリームを生成する符号化手段と、

上記ビデオデータに関して施された過去の符号化処理に関する過去の符号化パラメータを受け取り、上記過去の符号化パラメータを選択的に上記再符号化ストリームに重畳するとともに、上記再符号化ストリーム中に記述される上記過去の符号化パラメータのデータセットを示す情報を、上記再符号化ストリームに重畳する制御手段と

を備えたことを特徴とする符号化装置。

39. ビデオデータを符号化する符号化方法において、

上記ビデオデータに関して施された過去の符号化処理に関する過去の符号化パラメータを受け取るステップと、

上記ビデオデータを符号化し、再符号化ストリームを生成する符号化ステップと、

上記ビデオデータに関して施された過去の符号化処理に関する過去の符号化パラメータを受け取り、上記過去の符号化パラメータを選択的に上記再符号化ストリームに重畳するとともに、上記再符号化ストリーム中に記述される上記過去の符号化パラメータのデータセットを示す情報を、上記再符号化ストリームに重畳する制御ステップと

から構成される符号化方法。

40. ビデオデータを符号化する符号化装置において、

上記ビデオデータに関して施された過去の符号化処理に関する過去の符号化パラメータを受け取る手段と、

上記ビデオデータを符号化し、再符号化ストリームを生成する符号化手段と、

上記過去の符号化パラメータを上記再符号化ストリーム中に

history_stream()として重畳するとともに、上記再符号化ストリーム中に記述される上記過去の符号化パラメータに関する情報を、re_coding_stream_info()として上記再符号化ストリーム中に重畳する制御手段と

を備えたことを特徴とする符号化装置。

41. ビデオデータを符号化する符号化方法において、

上記ビデオデータに関して施された過去の符号化処理に関する過去の符号化パラメータを受け取るステップと、

上記ビデオデータを符号化し、再符号化ストリームを生成する符号化ステップと、

上記過去の符号化パラメータを上記再符号化ストリーム中に history_stream() として重畳するとともに、上記再符号化ストリーム中に記述される上記過去の符号化パラメータに関する情報を、re_coding_stream_info() として上記再符号化ストリーム中に重畳する制御ステップとから構成される符号化方法。

4 2. 符号化ストリームを復号化する復号化装置において、

上記符号化ストリームをデコードし、復号化ビデオデータを生成する復号化手段と、

上記符号化ストリームから、上記符号化ストリーム中に重畳されている過去の符号化パラメータのデータセットに関する情報を抽出する手段と、

上記データセットに関する情報に基づいて、上記符号化ストリームから、上記過去の符号化パラメータを抽出する手段と

を備えたことを特徴とする復号化装置。

4 3. 符号化ストリームを復号化する復号化方法において、

上記符号化ストリームをデコードし、復号化ビデオデータを生成する復号化ステップと、

上記符号化ストリームから、上記符号化ストリーム中に重畳されている過去の符号化パラメータのデータセットに関する情報を抽出するステップと、

上記データセットに関する情報に基づいて、上記符号化ストリームから、上記過去の符号化パラメータを抽出するステップと

から構成される復号化方法。

4 4. 符号化ストリームを復号化する復号化装置において、

上記符号化ストリームをデコードし、復号化ビデオデータを生成する復号化手段と、

上記符号化ストリームから、上記符号化ストリーム中に重畳されている過去の符号化パラメータを同定しているデータセットの情報を抽出する手段と、

上記データセットの情報に基づいて、上記符号化ストリームから、上記過去の符号化パラメータを抽出する手段と、

上記復号化ビデオデータと上記抽出した過去の符号化パラメータを伝送する伝送手段と、

を備えたことを特徴とする復号化装置。

4 5. 符号化ストリームを復号化する復号化方法において、

上記符号化ストリームをデコードし、復号化ビデオデータを生成する復号化ステップと、

上記符号化ストリームから、上記符号化ストリーム中に重畳されている過去の符号化パラメータを同定しているデータセットの情報を抽出するステップと、

上記データセットの情報に基づいて、上記符号化ストリームから、上記過去の符号化パラメータを抽出するステップと、

上記復号化ビデオデータと上記抽出した過去の符号化パラメータを伝送する伝送ステップと、

から構成される復号化方法。

4 6. 符号化ストリームを復号化する復号化装置において、

上記符号化ストリームをデコードし、復号化ビデオデータを生成する復号化手段と、

上記符号化ストリームから、上記符号化ストリーム中に
`re_coding_stream_info()`として記述されているフラグ又は／及びインジケータを抽出する手段と、

上記フラグ又は／及びインジケータに基づいて、上記符号化ストリームから、過去の符号化パラメータを抽出する手段と、

上記復号化ビデオデータと上記抽出した過去の符号化パラメータを伝送する伝送手段と、

を備えたことを特徴とする復号化装置。

47. 符号化ストリームを復号化する復号化方法において、

上記符号化ストリームをデコードし、復号化ビデオデータを生成する復号化ステップと、

上記符号化ストリームから、上記符号化ストリーム中に `re_coding_stream_info()` として記述されているフラグ又は／及びインジケータを抽出するステップと、

上記フラグ又は／及びインジケータに基づいて、上記符号化ストリームから、過去の符号化パラメータを抽出するステップと、

上記復号化ビデオデータと上記抽出した過去の符号化パラメータを伝送する伝送ステップと、

から構成される復号化方法。

48. 符号化ストリームを記録媒体に記録する記録装置において、

ソース符号化ストリームを復号し、ビデオデータを生成するとともに、上記ソース符号化ストリームから過去の符号化処理において生成された符号化パラメータを抽出する復号化手段と、

上記ビデオデータに対して再符号化処理を施すことによって、再符号化ビデオストリームを生成する符号化手段と、

上記過去の符号化パラメータに基づいて、上記再符号化処理を制御するとともに、上記過去の符号化パラメータを上記再符号化ストリーム中に選択的に記述する制御手段と、

上記再符号化ストリームを記録媒体に記録する手段と、

を含むことを特徴とする記録装置。

49. 符号化ストリームを記録媒体に記録する記録方法において、

ソース符号化ストリームを復号し、ビデオデータを生成するとともに、上記ソース符号化ストリームから過去の符号化処理において生成された符号化パラメータを抽出する復号化ステップと、

上記ビデオデータに対して再符号化処理を施すことによって、再符号化ビデオストリームを生成する符号化ステップと、

上記過去の符号化パラメータに基づいて、上記再符号化処理を制御するとともに、上記過去の符号化パラメータを上記再符号化ストリーム中に選択的に記述する制御ステップと、

上記再符号化ストリームを記録媒体に記録するステップと、
を含むことを特徴とする記録方法。

50. 符号化ストリームを記録媒体に記録する記録装置において、

供給されたビデオデータに関して施された過去の符号化処理に関する過去の符号化パラメータを受け取る手段と、

上記ビデオデータを符号化し、再符号化ストリームを生成する符号化手段と、

上記過去の符号化パラメータを上記再符号化ストリーム中に
history_stream()として記述するとともに、上記再符号化ストリーム中に記述される上記過去の符号化パラメータに関する情報を、re_coding_stream_info()として上記再符号化ストリーム中に記述する制御手段と

上記再符号化ストリームを記録媒体に記録する記録手段と
を備えたことを特徴とする記録装置。

51. 符号化ストリームを記録媒体に記録する記録方法において、

供給されたビデオデータに関して施された過去の符号化処理に関する過去の符号化パラメータを受け取るステップと、

上記ビデオデータを符号化し、再符号化ストリームを生成する符号化ステップと、

上記過去の符号化パラメータを上記再符号化ストリーム中に `history_stream()` として記述するとともに、上記再符号化ストリーム中に記述される上記過去の符号化パラメータに関する情報を、`re_coding_stream_info()` として上記再符号化ストリーム中に記述する制御ステップと

上記再符号化ストリームを記録媒体に記録する記録ステップと
を備えたことを特徴とする記録方法。

52. 符号化ストリームを記録媒体から再生する再生装置において、

上記記録媒体からソース符号化ストリームを再生する再生手段と、

上記ソース符号化ストリームを復号し、ビデオデータを生成するとともに、上記ソース符号化ストリームから過去の符号化処理において生成された符号化パラメータを抽出する復号化手段と、

上記ビデオデータに対して再符号化処理を施すことによって、再符号化ビデオストリームを生成する符号化手段と、

上記過去の符号化パラメータに基づいて、上記再符号化処理を制御するとともに、上記過去の符号化パラメータを上記再符号化ストリーム中に選択的に記述する制御手段と、

を含むことを特徴とする再生装置。

53. 符号化ストリームを記録媒体から再生する再生方法において、

上記記録媒体からソース符号化ストリームを再生する再生ステップと、

上記ソース符号化ストリームを復号し、ビデオデータを生成するとともに、上記ソース符号化ストリームから過去の符号化処理において生成された符号化パラメータを抽出する復号化ステップと、

上記ビデオデータに対して再符号化処理を施すことによって、再符号化ビデオ

ストリームを生成する符号化ステップと、

上記過去の符号化パラメータに基づいて、上記再符号化処理を制御するとともに、上記過去の符号化パラメータを上記再符号化ストリーム中に選択的に記述する制御ステップと、

を含むことを特徴とする再生方法。

5 4. 符号化ストリームを記録媒体から再生する再生装置において、

上記記録媒体から符号化ストリームを再生する手段と、

上記符号化ストリームをデコードし、復号化ビデオデータを生成する復号化手段と、

上記符号化ストリームから、上記符号化ストリーム中に
re_coding_stream_info()として記述されているフラグ又は／及びインジケータ
を抽出する手段と、

上記フラグ又は／及びインジケータに基づいて、上記符号化ストリームから、
過去の符号化パラメータを抽出する手段と、

上記復号化ビデオデータと上記抽出した過去の符号化パラメータを伝送する伝
送手段と、

を備えたことを特徴とする再生装置。

5 5. 符号化ストリームを記録媒体から再生する再生方法において、

上記記録媒体から符号化ストリームを再生するステップと、

上記符号化ストリームをデコードし、復号化ビデオデータを生成する復号化ス
テップと、

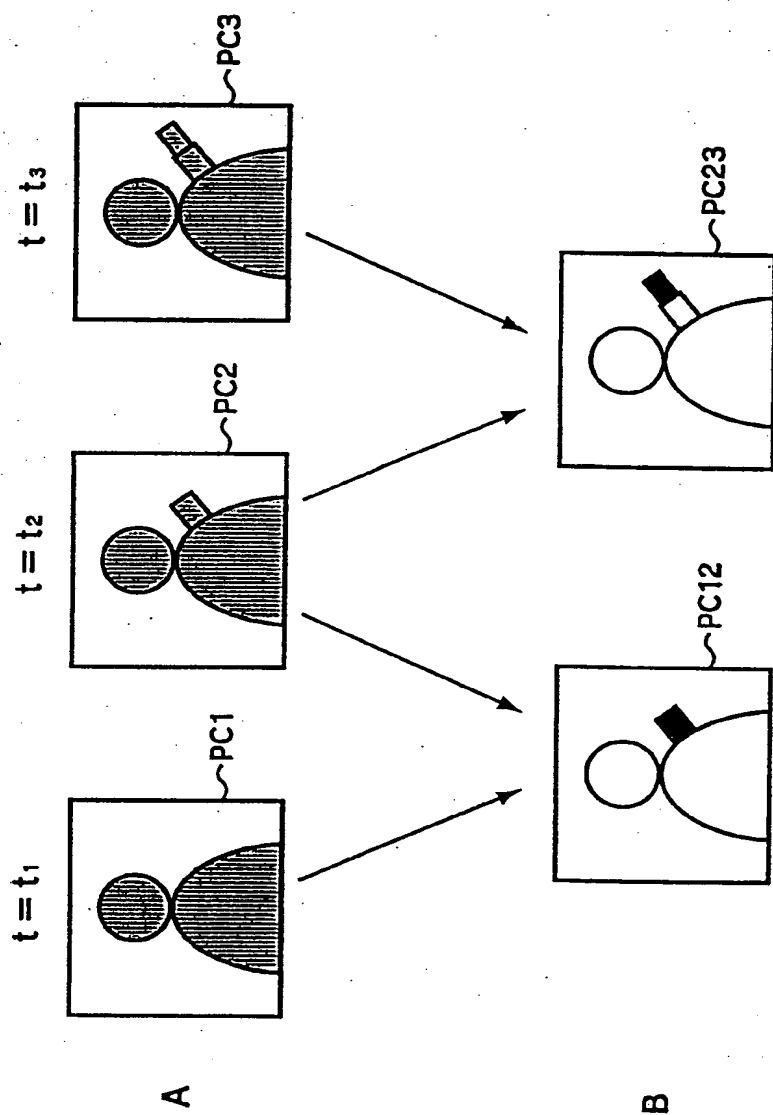
上記符号化ストリームから、上記符号化ストリーム中に
re_coding_stream_info()として記述されているフラグ又は／及びインジケータ
を抽出するステップと、

上記フラグ又は／及びインジケータに基づいて、上記符号化ストリームから、

過去の符号化パラメータを抽出するステップと、

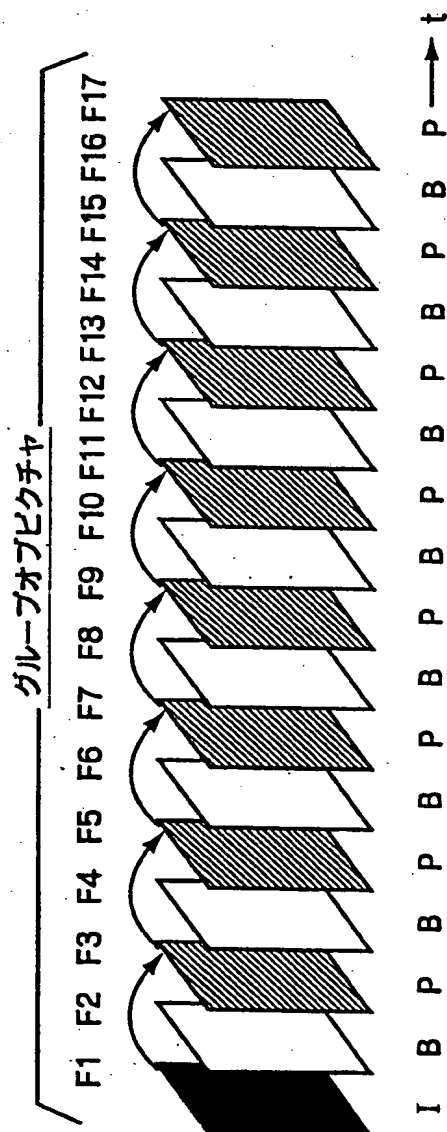
上記復号化ビデオデータと上記抽出した過去の符号化パラメータを伝送する伝送ステップと、

から構成される再生方法。



高能率符号化

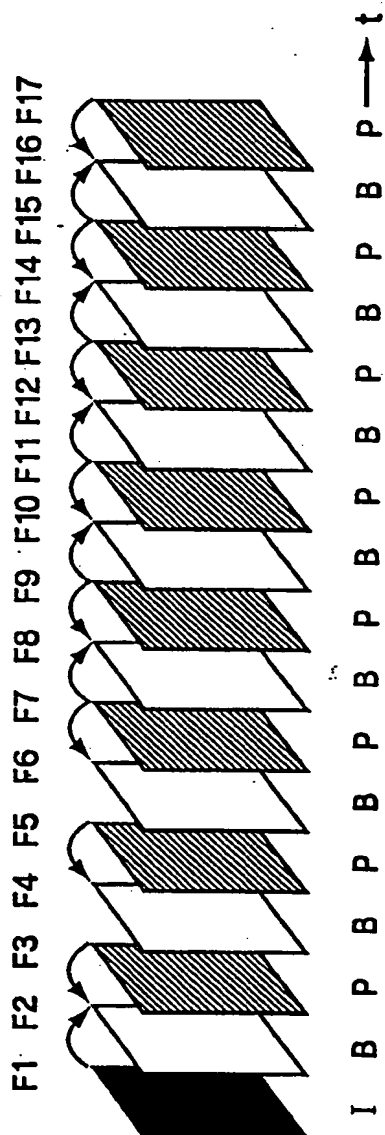
第1図



P-ピクチャ (前方向予測)

ピクチャタイプ I, P, B-picture

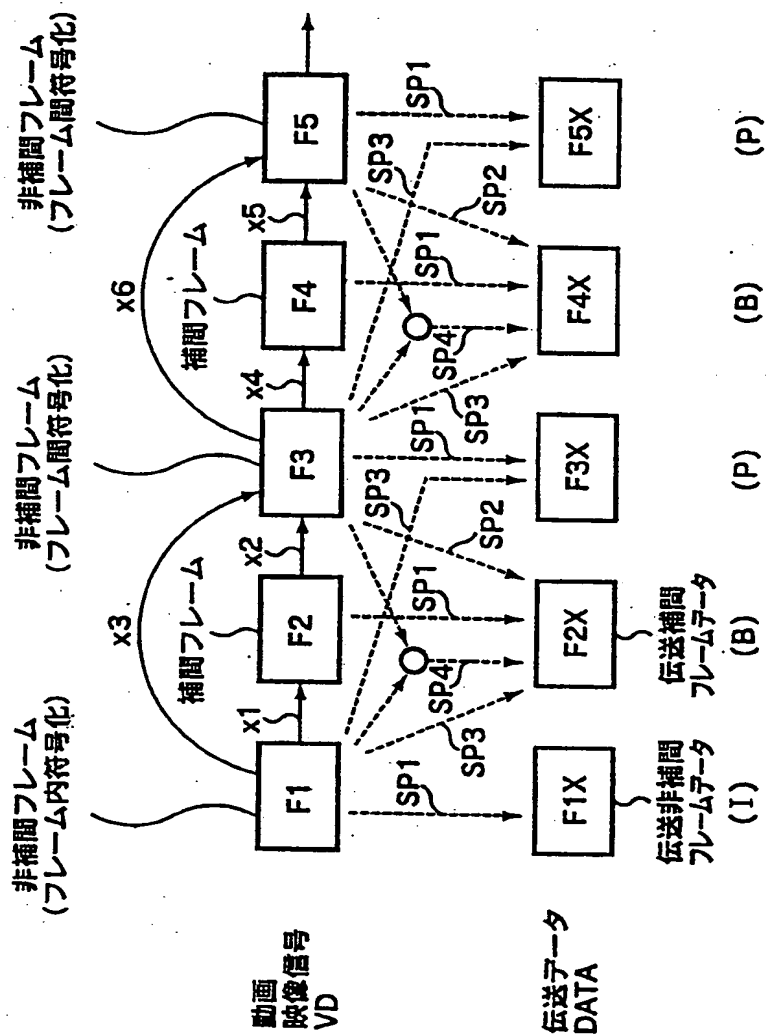
图2 续



B-ピクチャ (両方向予測)

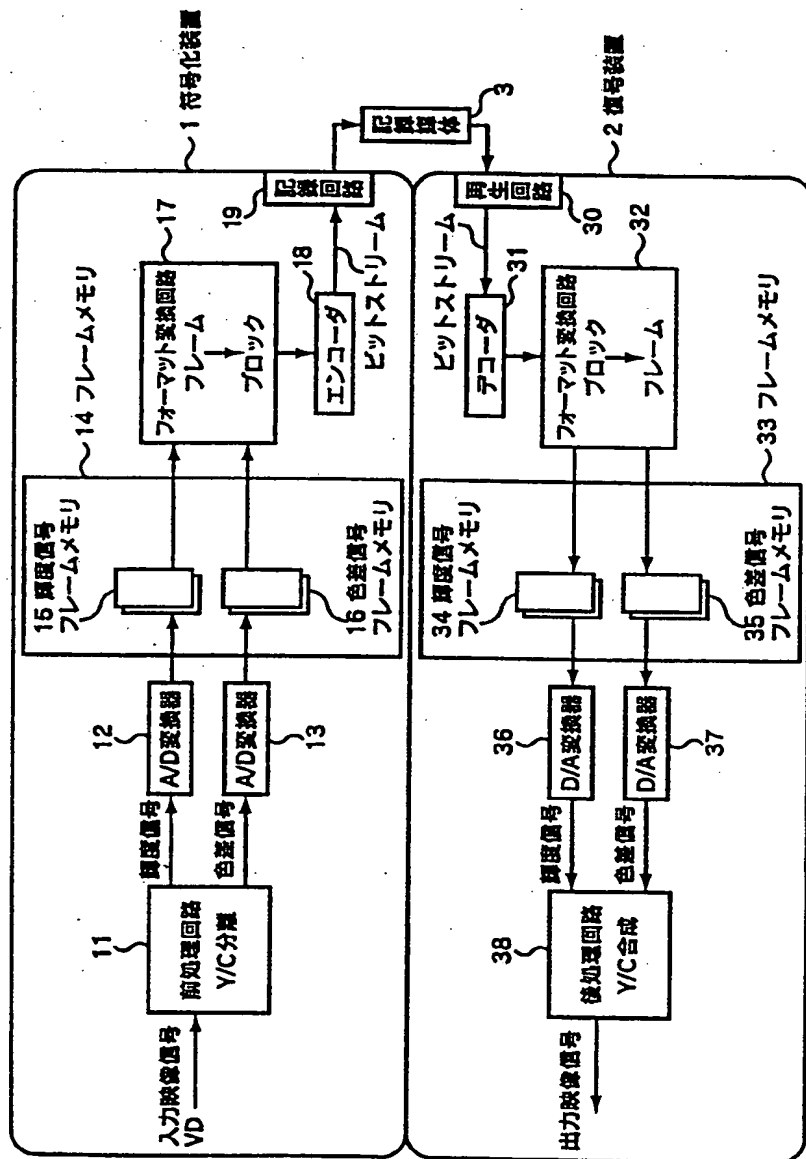
ピクチャタイプ I, P, B-picture

第3図



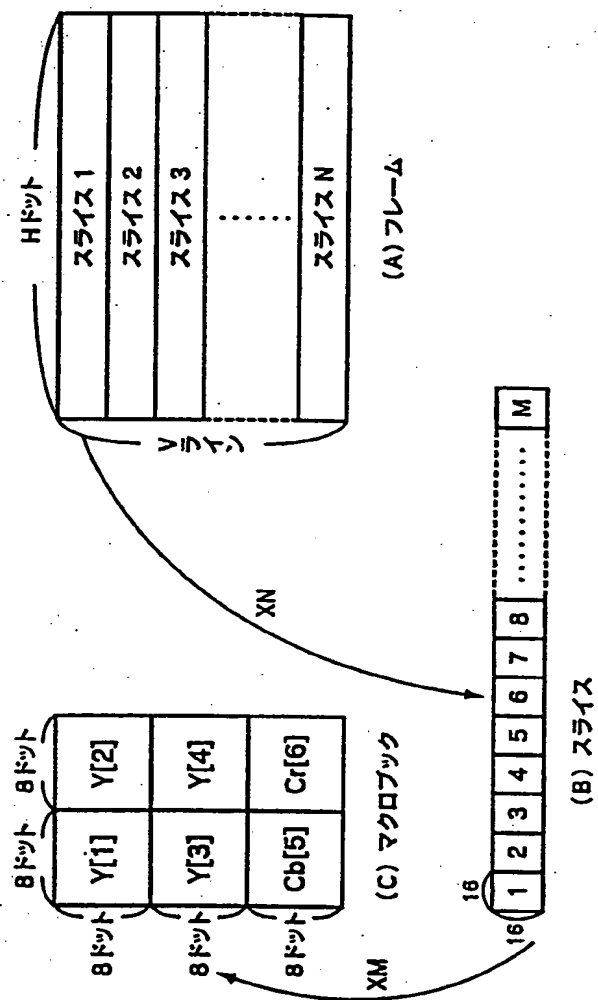
動画信号符号化方法の原理

第4図



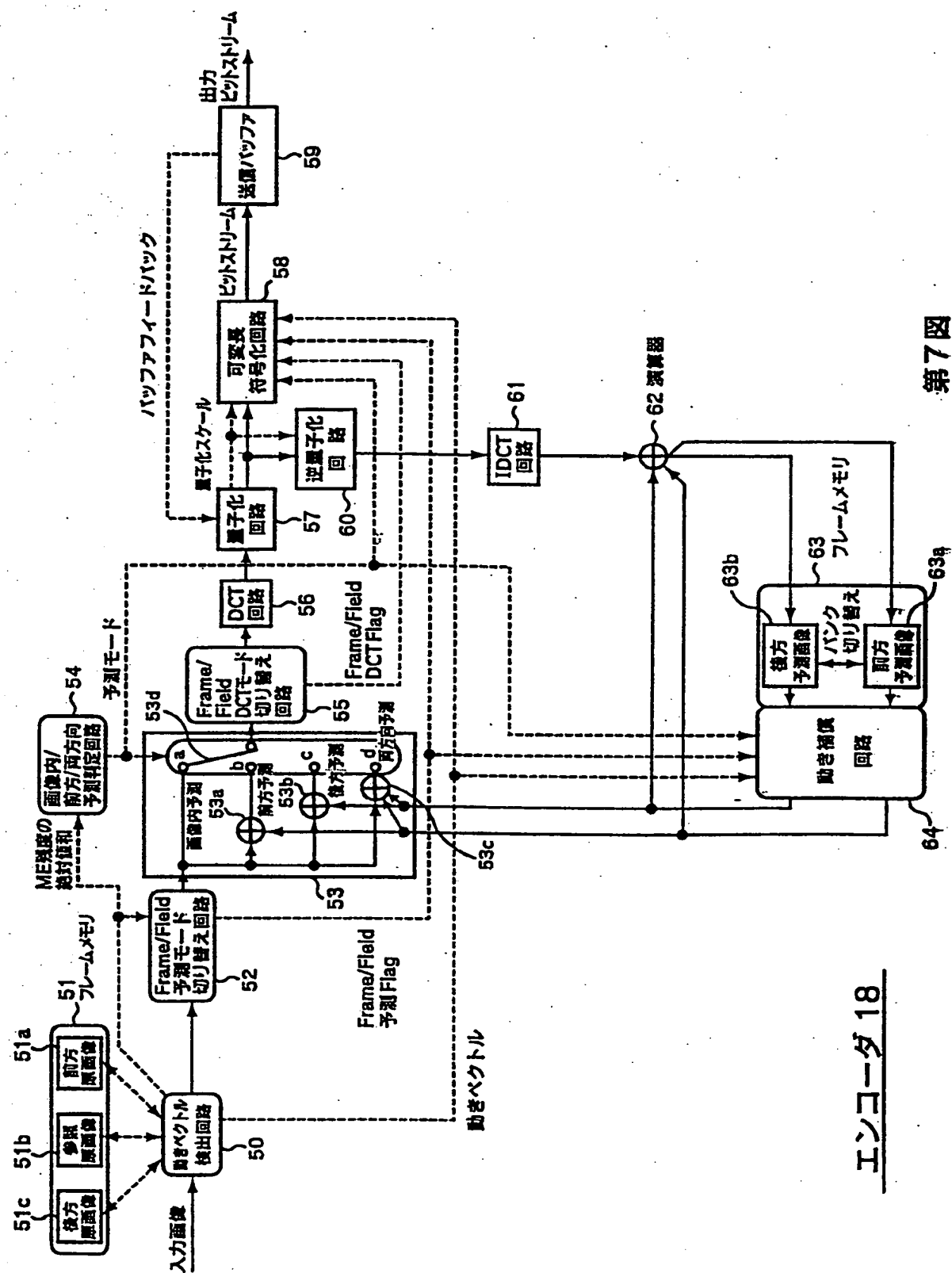
動画像符号化/復号装置

第5図

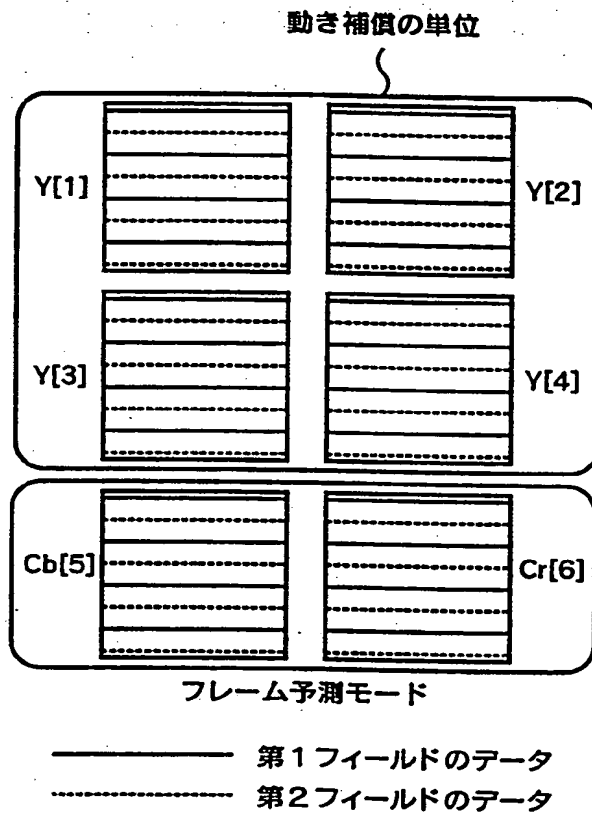


画像データの構造

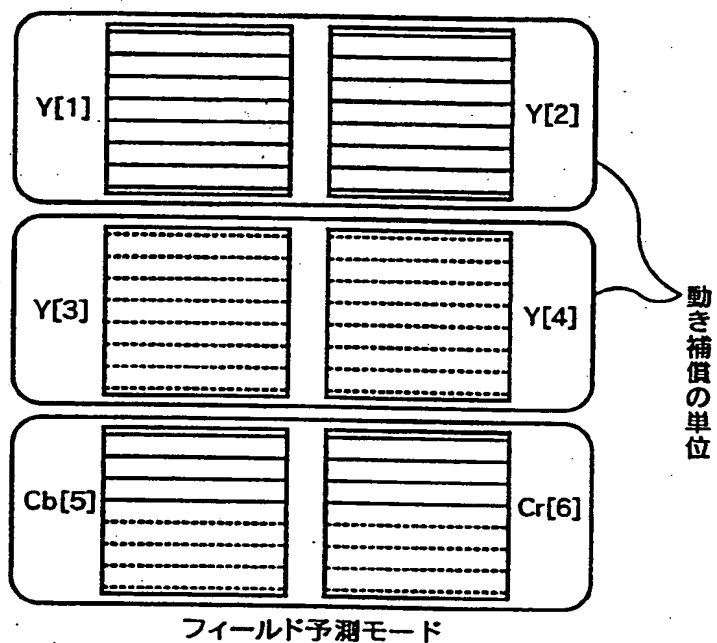
第6図



第七圖

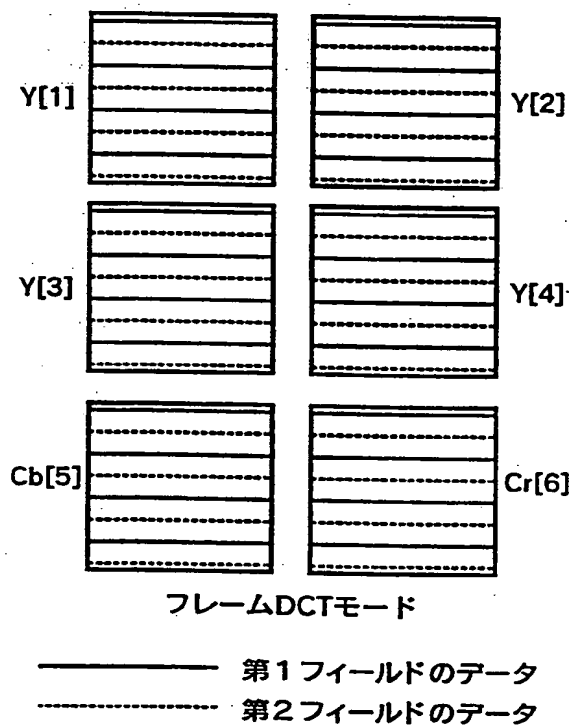


第8図



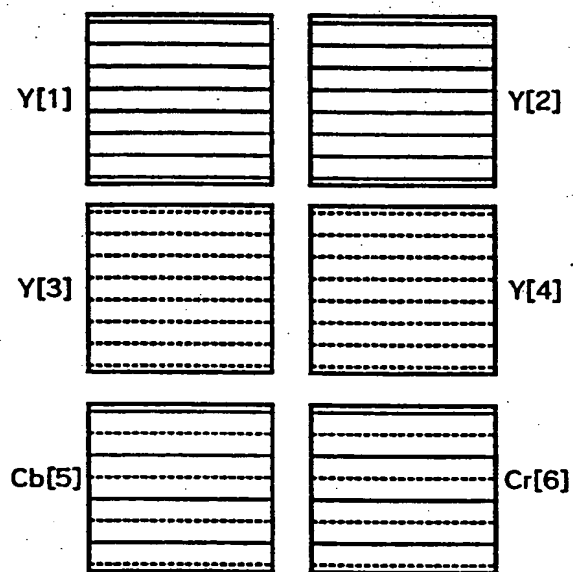
—— 第1フィールドのデータ
----- 第2フィールドのデータ

第9図



フレームDCTモード

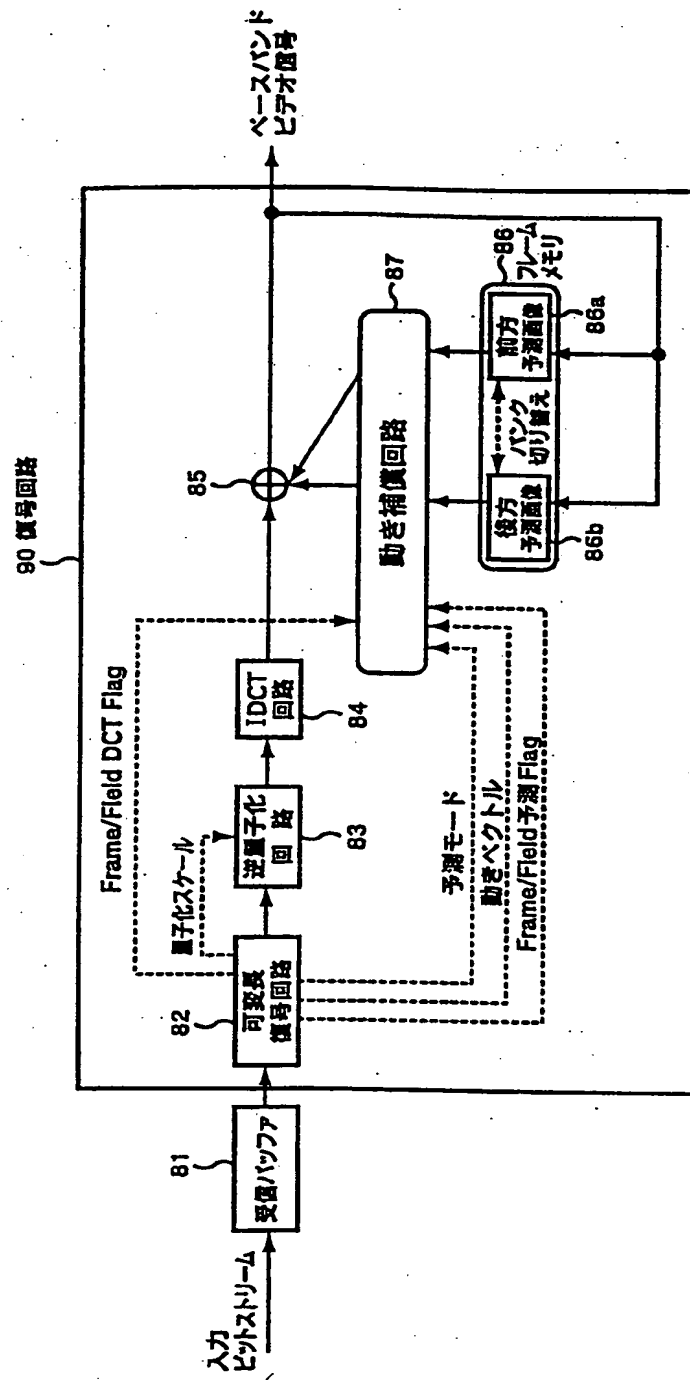
第10図



フィールドDCTモード

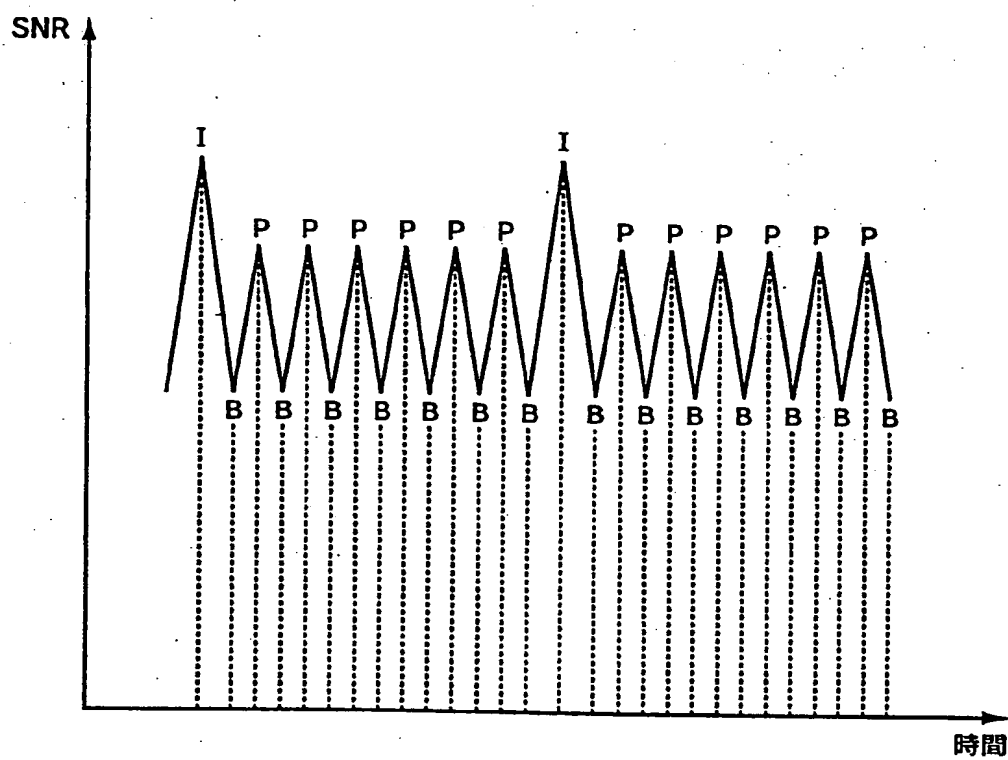
—— 第1フィールドのデータ
----- 第2フィールドのデータ

第11図

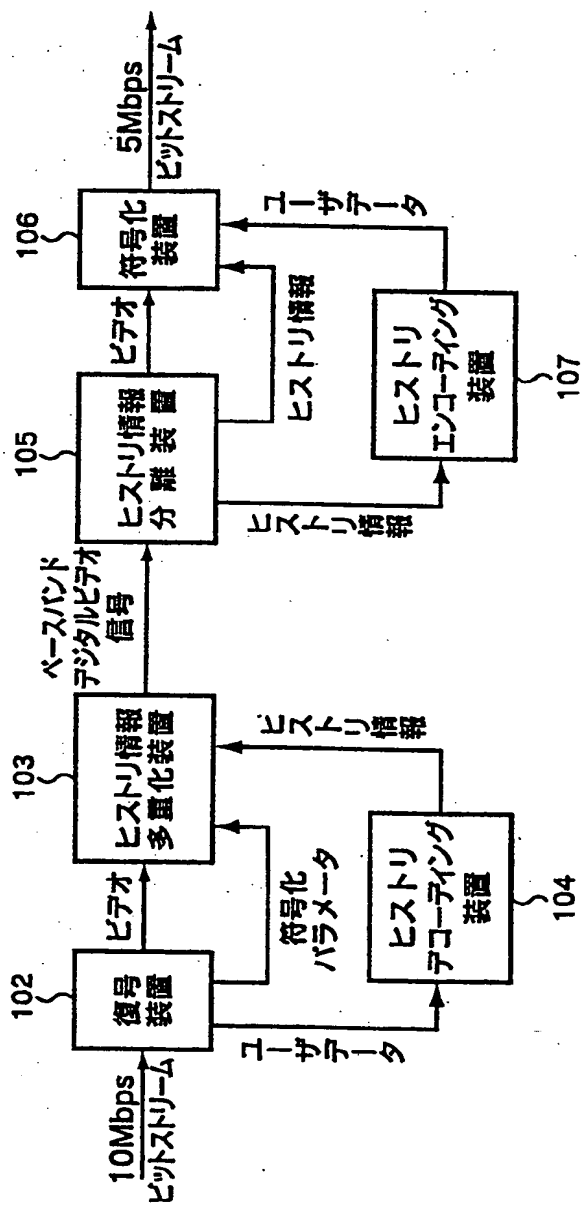


デコーダ 31

第12図



第13図



トランスコーダ 101

第14図

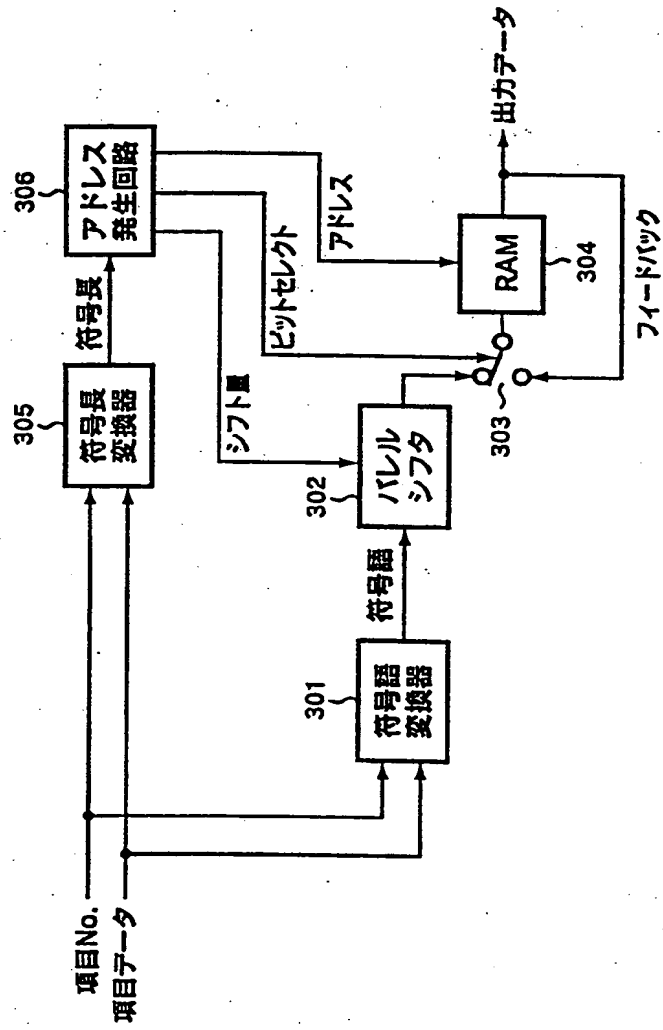
0	1	2	13	14	15
16	17	18	29	30	31
32	33	34	45	46	47
⋮	⋮	⋮		⋮	⋮	⋮
⋮	⋮	⋮		⋮	⋮	⋮
⋮	⋮	⋮		⋮	⋮	⋮
⋮	⋮	⋮		⋮	⋮	⋮
⋮	⋮	⋮		⋮	⋮	⋮
⋮	⋮	⋮		⋮	⋮	⋮
⋮	⋮	⋮		⋮	⋮	⋮
208	209	210	221	222	223
224	225	226	237	238	239
240	241	242	251	254	255

マクロブロック

第 17 図

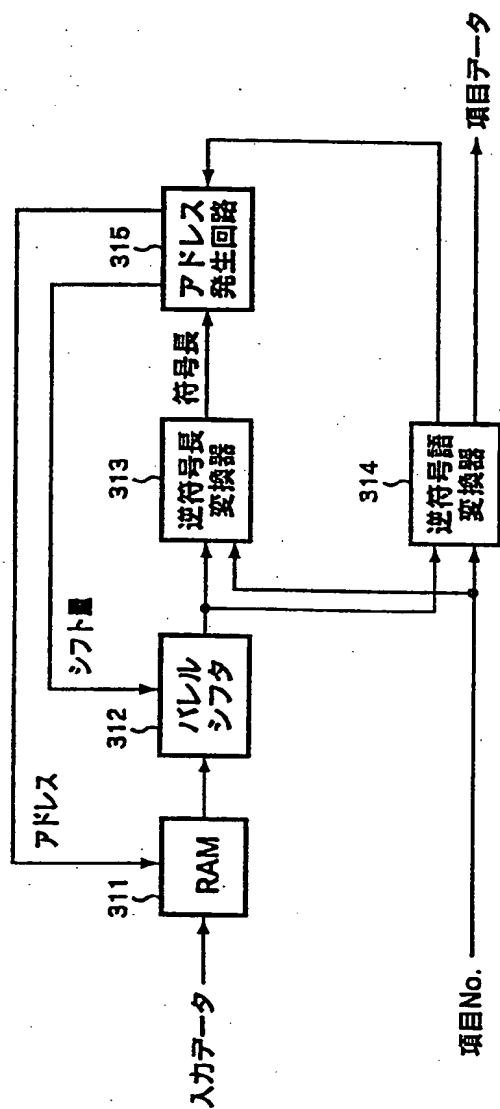
画像データ領域									ヒストリ情報領域			
	第 1 世代			第 2 世代			第 3 世代					
D9	Cb[0][9]	Y[0][9]	Cr[0][9]	Y[1][9]	Cb[1][9]	Y[2][9]	Cr[1][9]	Y[3][9]				
D8	Cb[0][8]	Y[0][8]	Cr[0][8]	Y[1][8]	Cb[1][8]	Y[2][8]	Cr[1][8]	Y[3][8]				
D7	Cb[0][7]	Y[0][7]	Cr[0][7]	Y[1][7]	Cb[1][7]	Y[2][7]	Cr[1][7]	Y[3][7]				
D6	Cb[0][6]	Y[0][6]	Cr[0][6]	Y[1][6]	Cb[1][6]	Y[2][6]	Cr[1][6]	Y[3][6]				
D5	Cb[0][5]	Y[0][5]	Cr[0][5]	Y[1][5]	Cb[1][5]	Y[2][5]	Cr[1][5]	Y[3][5]				
D4	Cb[0][4]	Y[0][4]	Cr[0][4]	Y[1][4]	Cb[1][4]	Y[2][4]	Cr[1][4]	Y[3][4]				
D3	Cb[0][3]	Y[0][3]	Cr[0][3]	Y[1][3]	Cb[1][3]	Y[2][3]	Cr[1][3]	Y[3][3]				
D2	Cb[0][2]	Y[0][2]	Cr[0][2]	Y[1][2]	Cb[1][2]	Y[2][2]	Cr[1][2]	Y[3][2]				
D1												
D0	Cb[0][x]	Y[0][x]	Cr[0][x]	Y[1][x]	Cb[1][x]	Y[2][x]	Cr[1][x]	Y[3][x]				

第18図

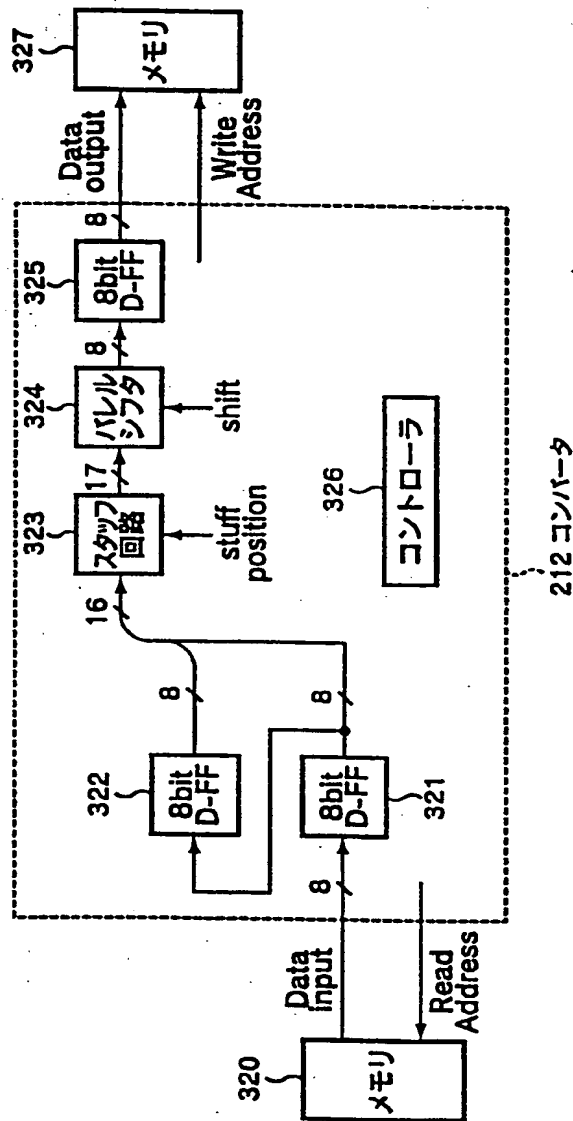


ヒストリVLC 211

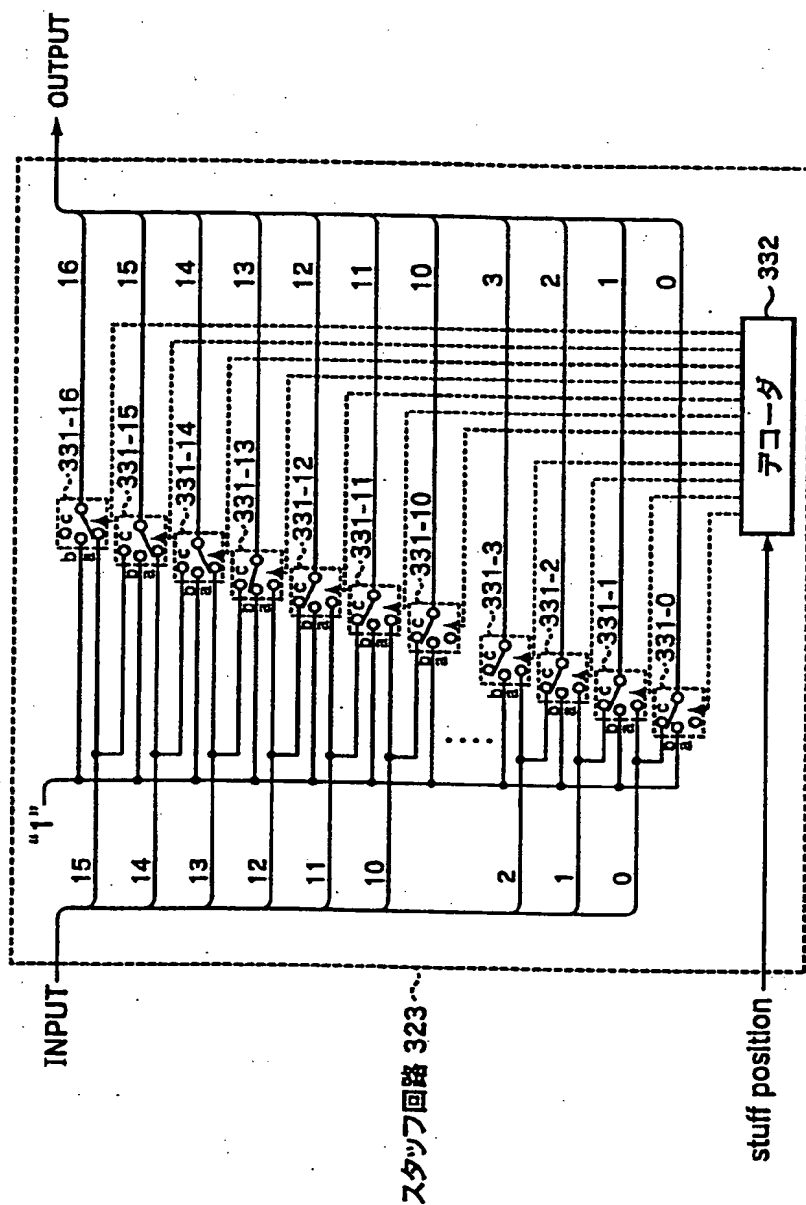
第20図

ヒストリVLD 203

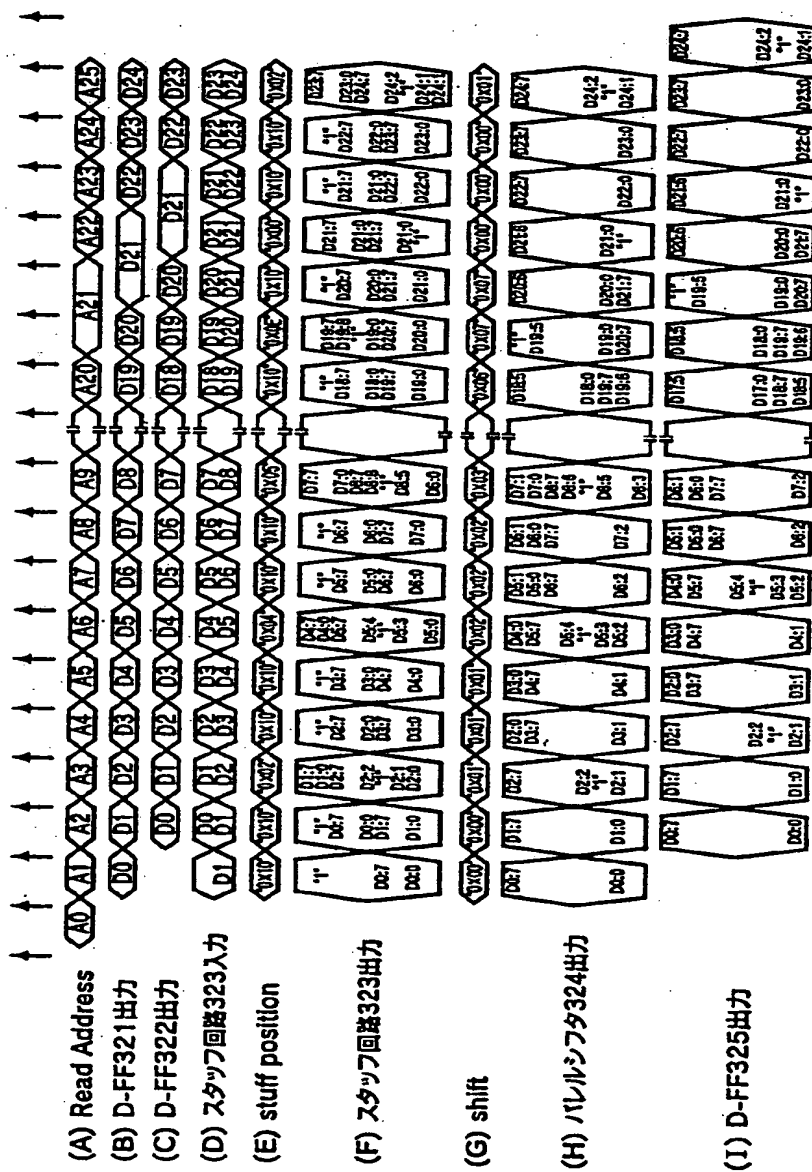
第21図



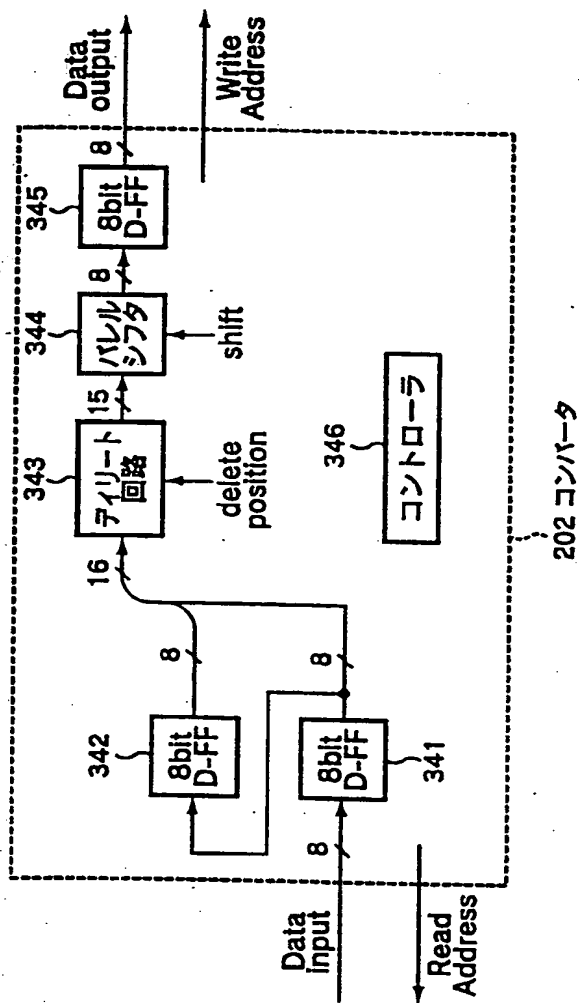
第22図



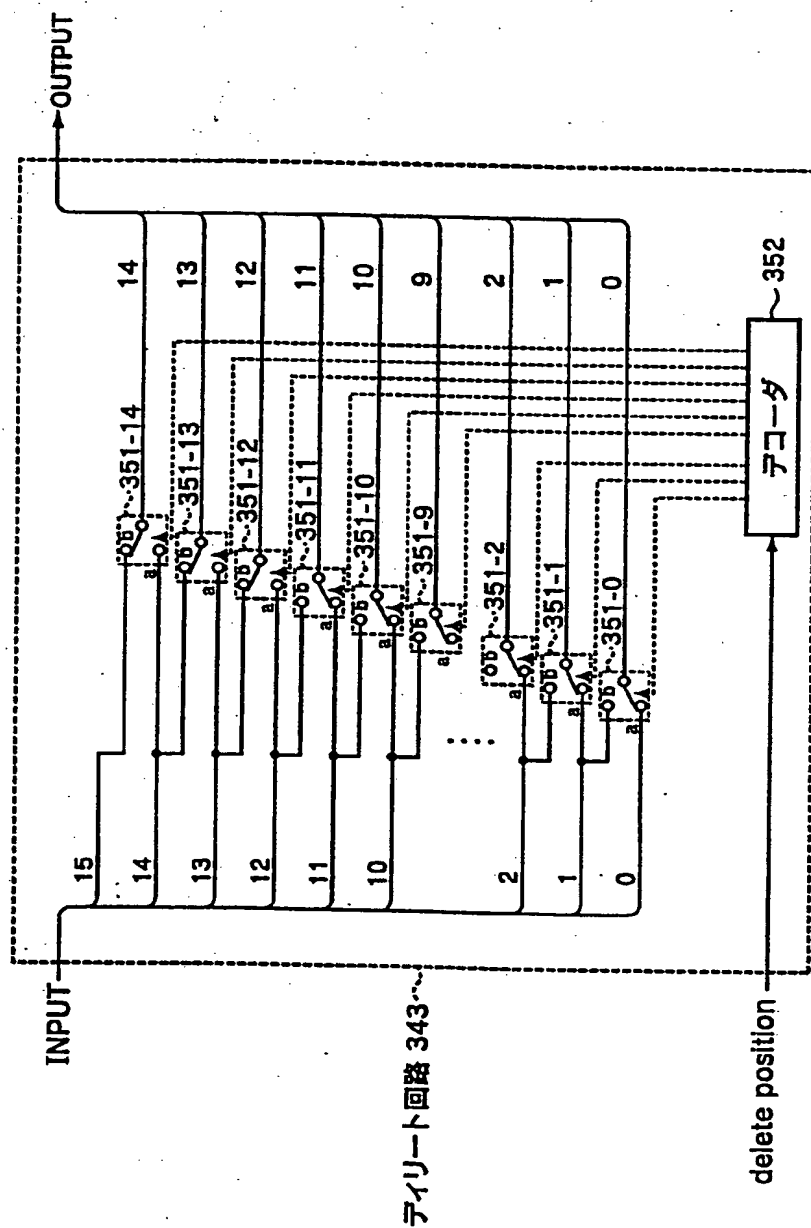
第23図



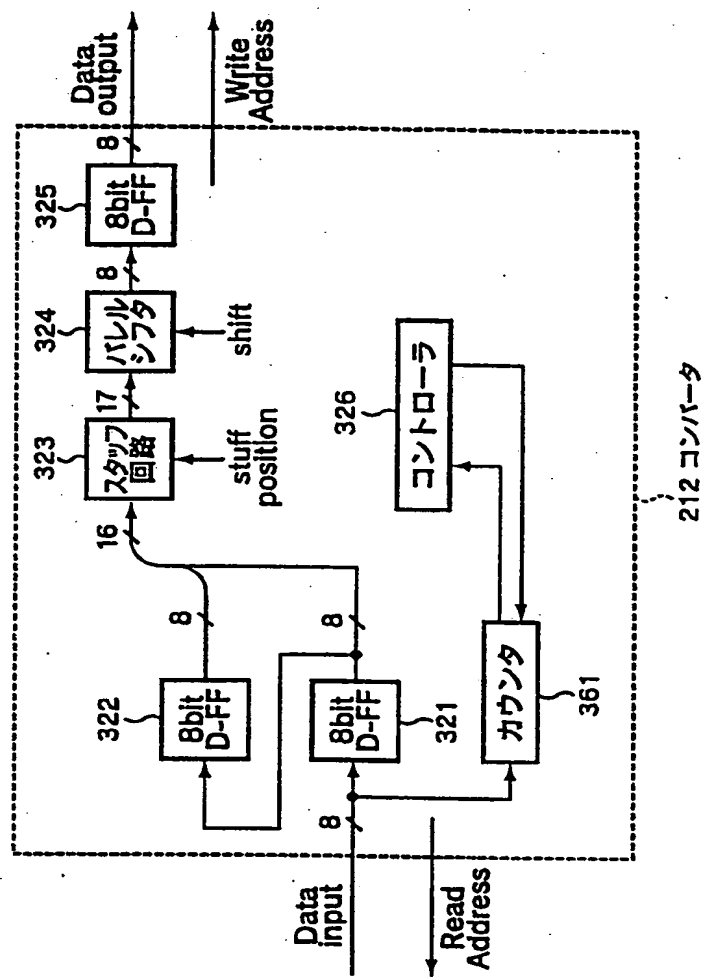
第24図



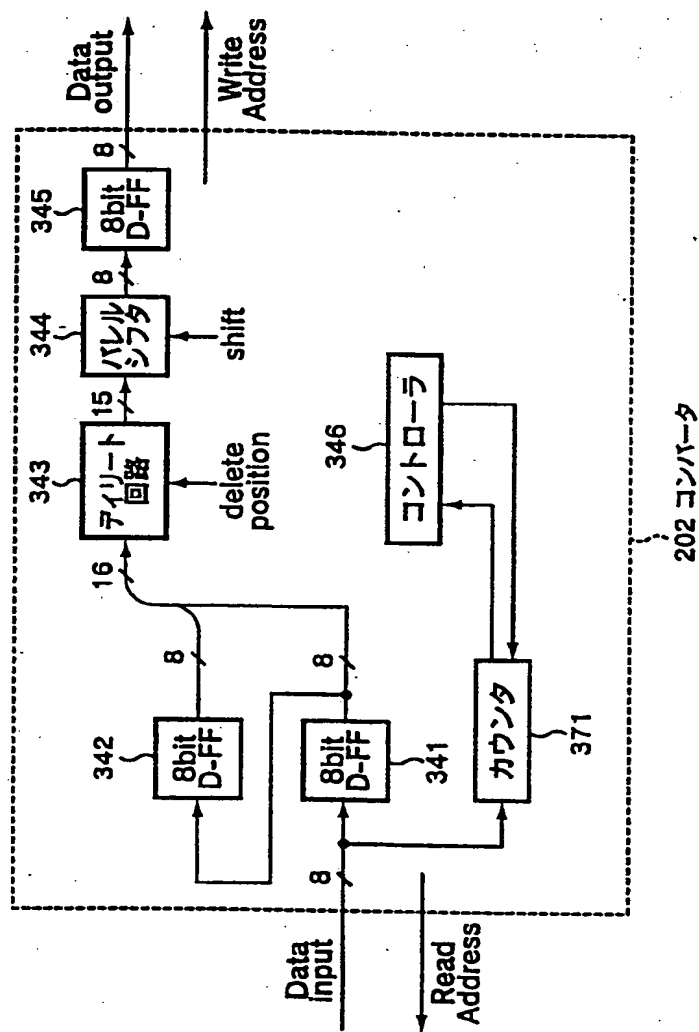
第25図



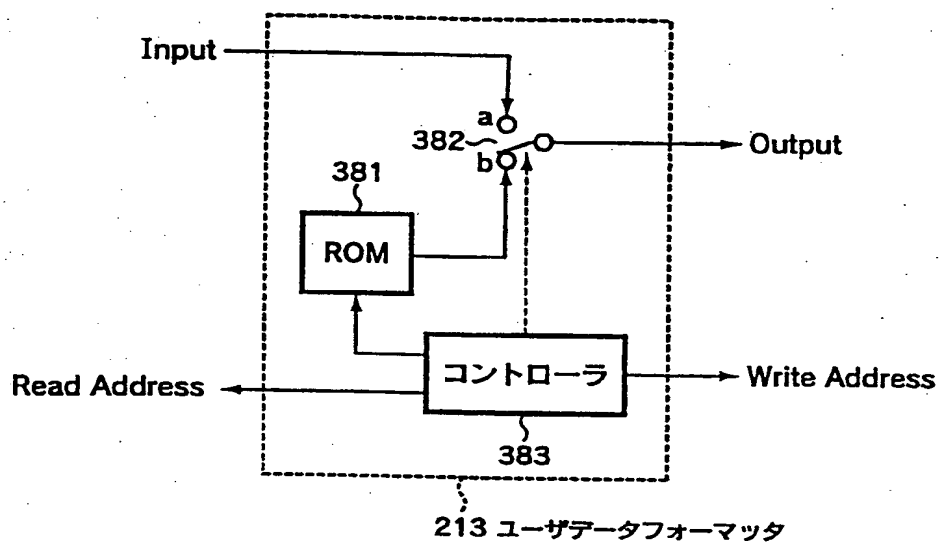
第26図



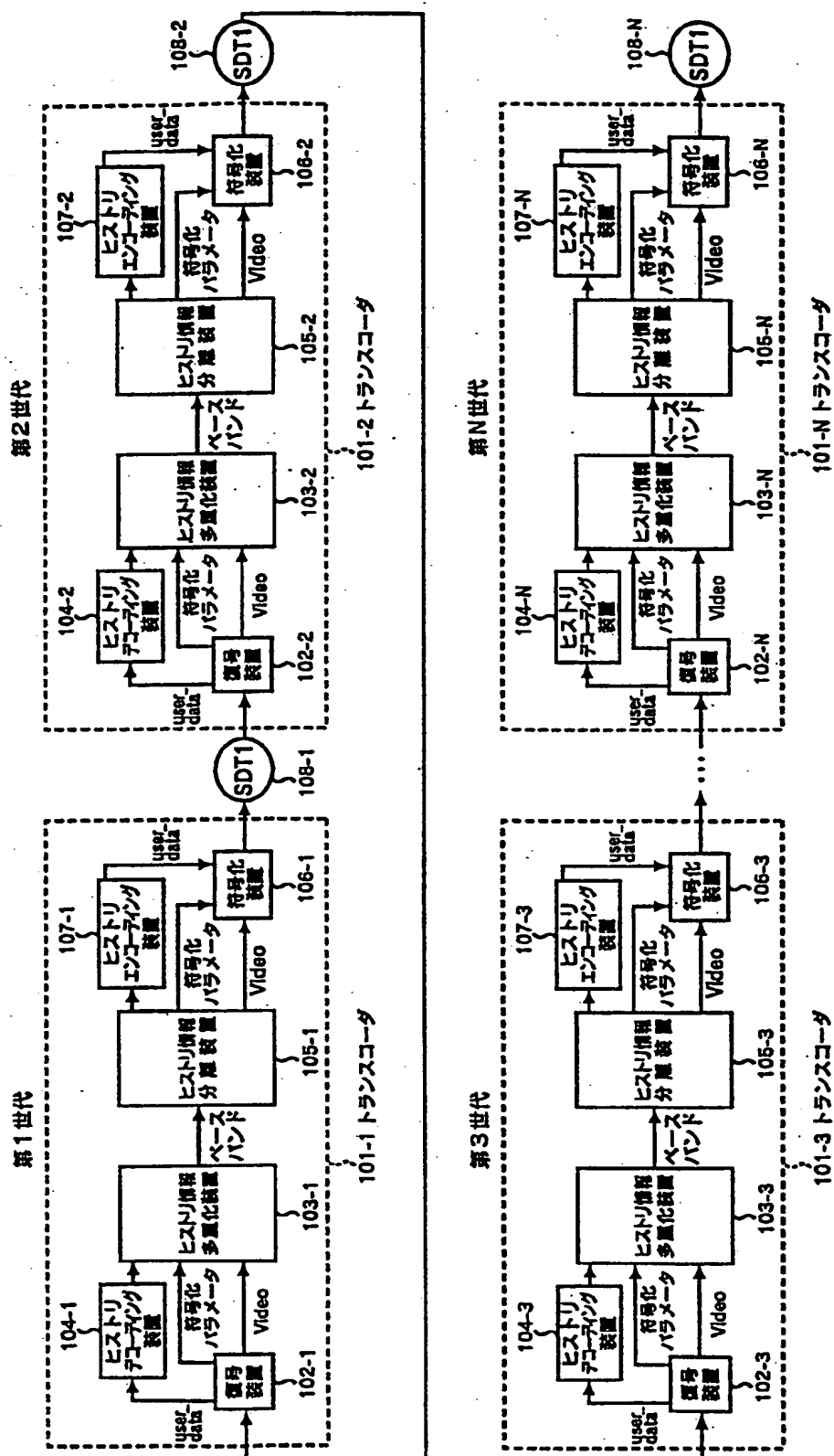
第27図



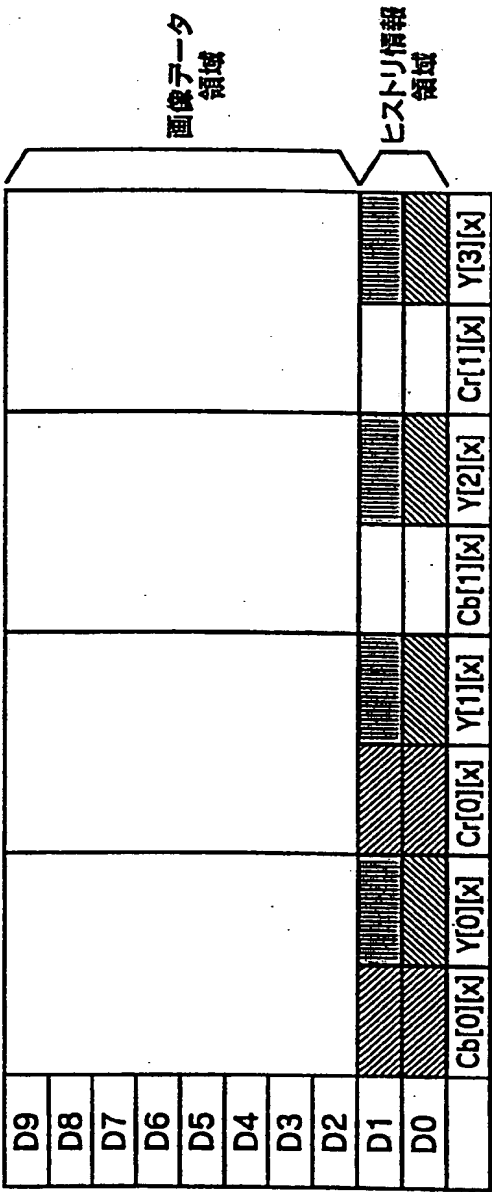
第28図



第29図

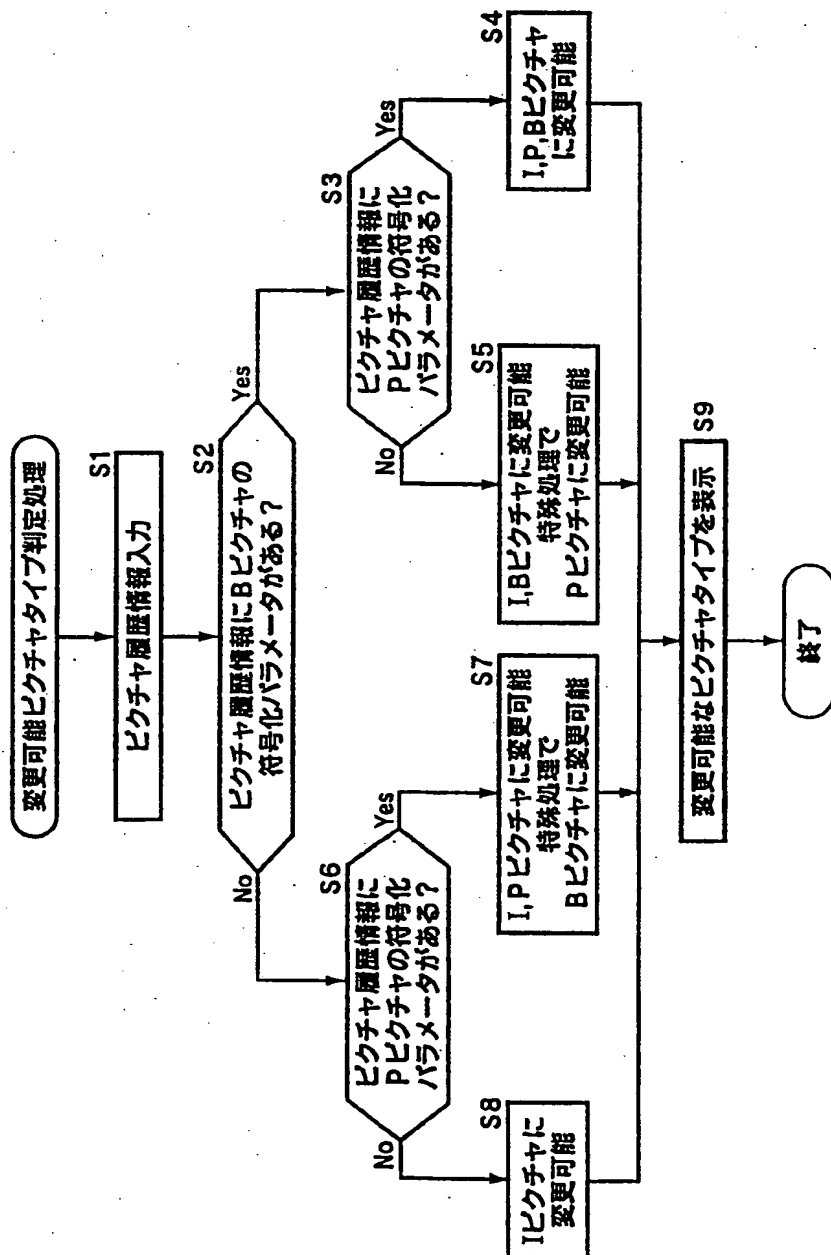


第30図



第31図

- Iピクチャの符号化パラメータ
- Pピクチャの符号化パラメータ
- Bピクチャの符号化パラメータ



第32図

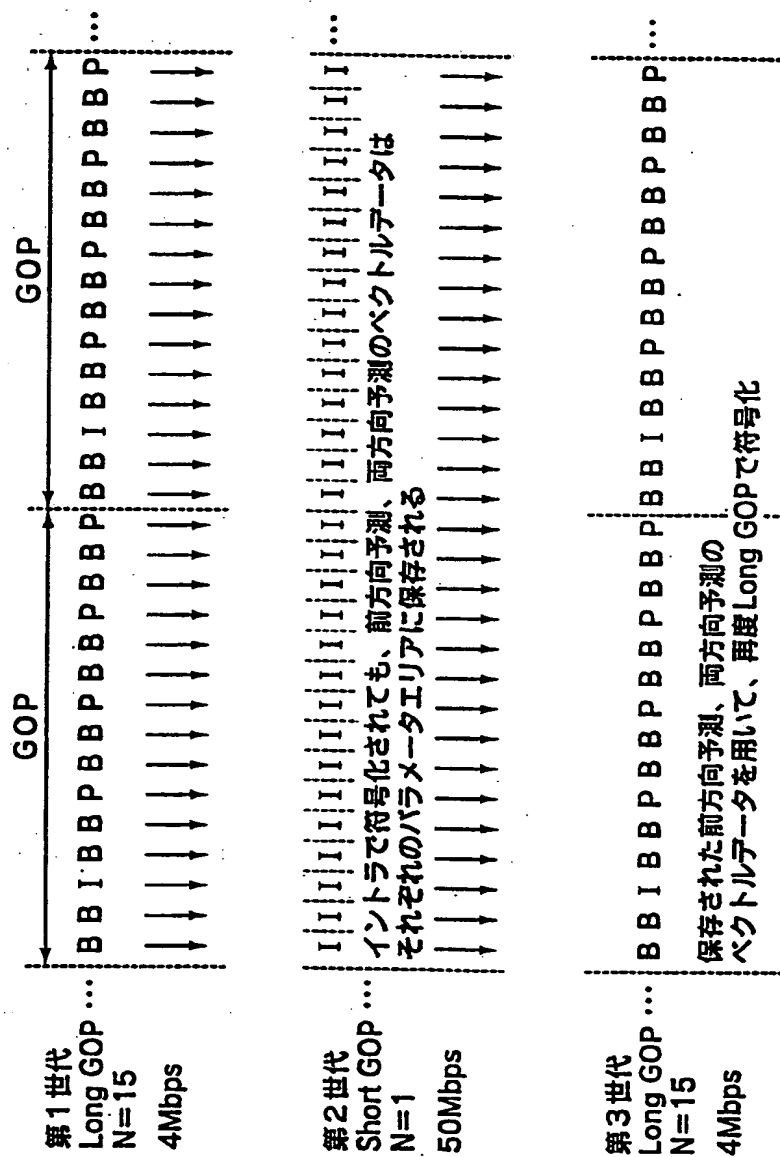
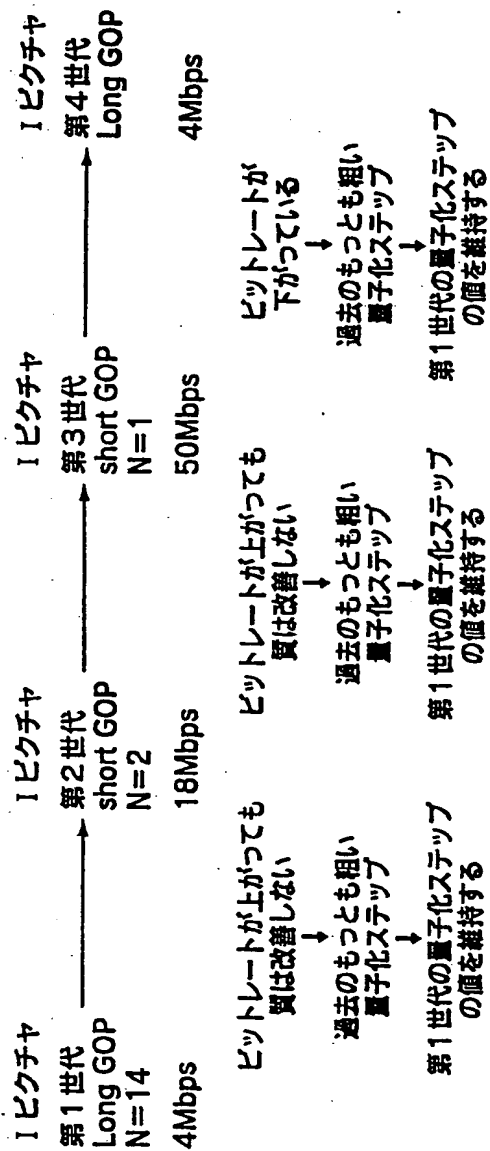
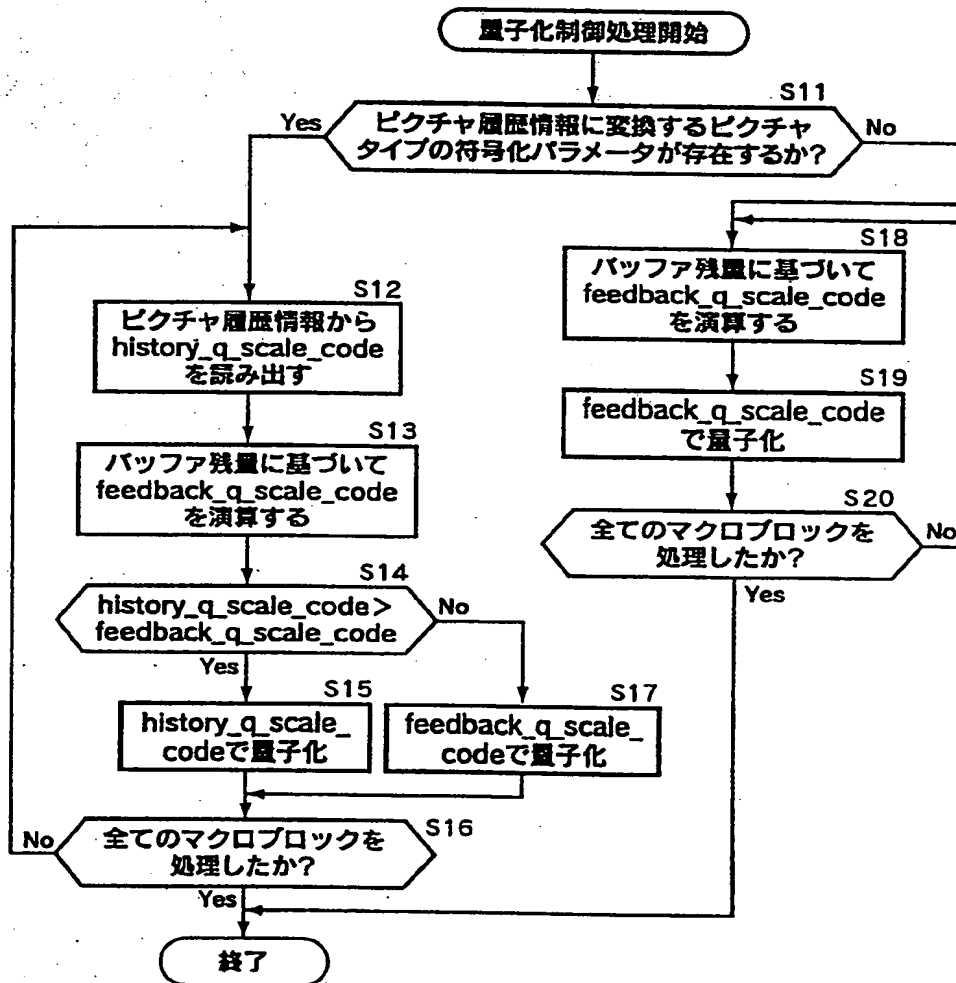


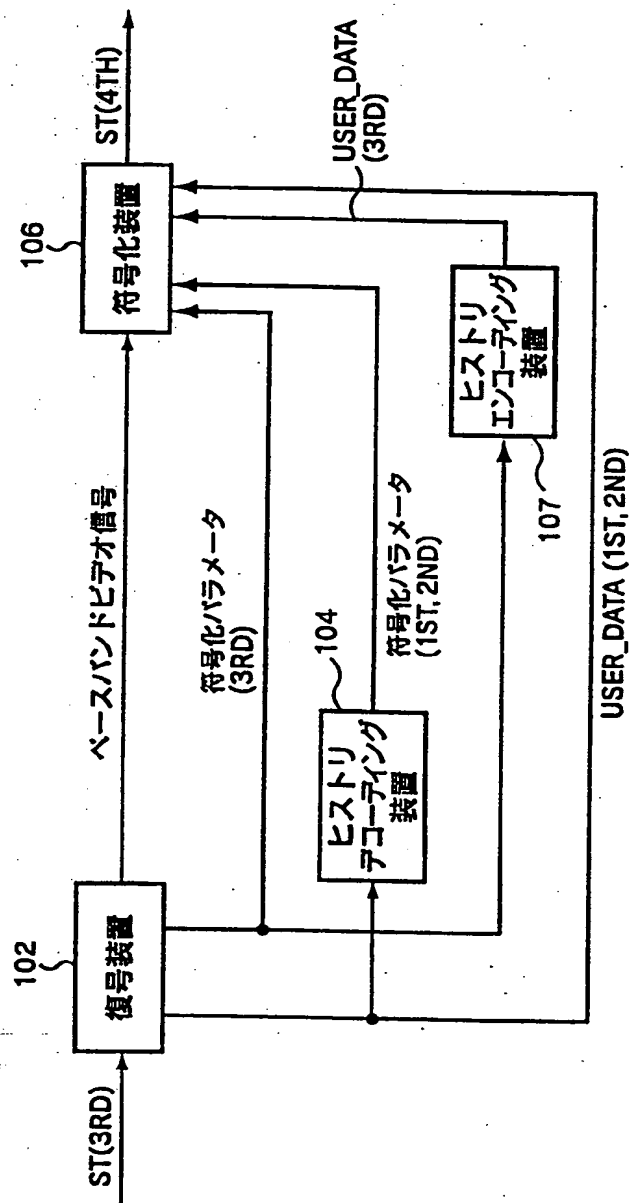
圖
三
張



第35図



第36図

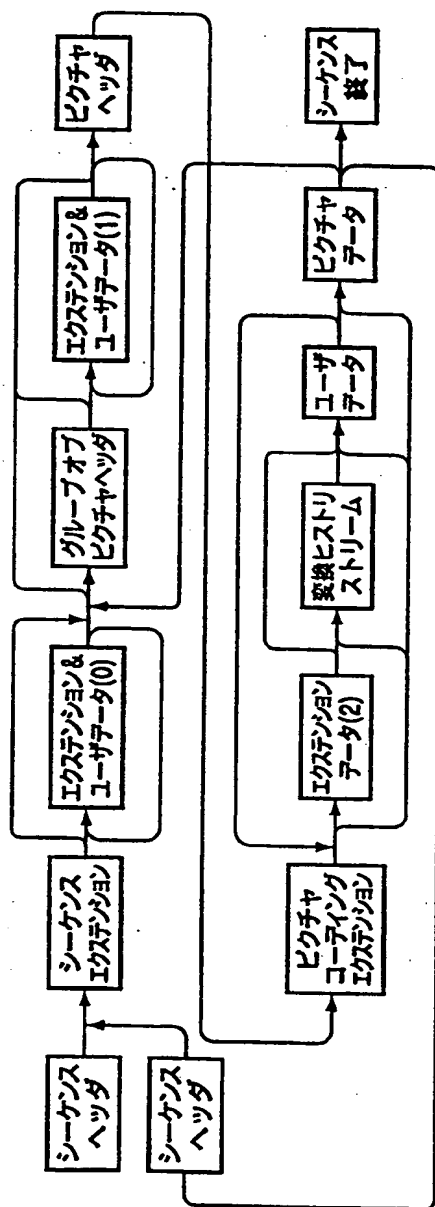


第37図

stream with history data

video_sequence () {	No. of bits	Mnemonic
next_start_code ()		
sequence_header ()		
sequence_extension ()		
do {		
extension_and_user_data (0)		
do {		
if(nextbits() == group_start_code){		
group_of_pictures_header(1)		
extension_and_user_data (1)		
}		
picture_header()		
picture_coding_extension()		
while((nextbits() == extension_start_code)		
(nextbits() == user_data_start_code)){		
if(nextbits() == extension_start_code){		
extension_data(2)		
if(nextbits() == user_data_start_code){		
user_data_start_code	32	bslbf
if(nextbits() == History_Data_ID){		
History_Data_ID	8	bslbf
converted_history_stream()		
}		
else{		
user_data()		
}.		
}		
}		
picture_data()		
}while((nextbits() == picture_start_code)		
(nextbits() == groupe_start_code))		
if(nextbits() != sequence_end_code){		
sequence_header ()		
sequence_extension ()		
}		
}while(nextbits() != sequence_end_code)		
sequence_end_code		
}	32	bslbf

第38図



第39図

history stream(40-1)

history_stream(){	bits	value
sequence_header		
sequence_header_code	32	000001B3
sequence_header_present_flag	1	
horizontal_size_value	12	
marker_bit	1	1
vertical_size_value	12	
aspect_ratio_information	4	
frame_rate_code	4	
marker_bit	1	1
bit_rate_value	18	
marker_bit	1	1
vbv_buffer_size_value	10	
constrained_parameter_flag	1	0
load_intra_quantiser_matrix	1	
load_non_intra_quantiser_matrix	1	
marker_bits	5	1F
intra_quantiser_matrix[64]	8*64	
non_intra_quantiser_matrix[64]	8*64	
sequence_extension		
extension_start_code	32	000001B5
extension_start_code_identifier	4	1
sequence_extension_present_flag	1	
profile_and_level_indication	8	
progressive_sequence	1	
chroma_format	2	
horizontal_size_extension	2	
vertical_size_extension	2	
marker_bit	1	1
bit_rate_extension	12	
vbv_buffer_size_extension	8	
low_delay	1	
marker_bit	1	1

第40図

history stream(40-2)

	bits	value
frame_rate_extension_n	2	
frame_rate_extension_d	5	
marker_bits	6	3F
sequence_display_extension		
extension_start_code	32	000001B5
extension_start_code_identifier	4	2
sequence_display_extension_present_flag	1	
video_format	3	
colour_description	1	
colour_primaries	8	
transfer_characteristics	8	
marker_bit	1	1
matrix_coefficients	8	
display_horizontal_size	14	
marker_bit	1	1
display_vertical_size	14	
marker_bit	1	1
macroblock_assignment_in_user_data		
macroblock_assignment_present_flag	1	
marker_bit	7	7F
v_phase	8	
h_phase	8	
group_of_picture_header		
group_start_code	32	000001B8
group_of_picture_header_present_flag	1	
time_code	25	
closed_gop	1	
broken_link	1	
marker_bits	4	F
picture_header		
picture_start_code	32	00000100

history stream(40-3)

	bits	value
temporal_reference	10	
picture_coding_type	3	
marker_bit	1	1
vbv_delay	16	
full_pel_forward_vector	1	
forward_f_code	3	
full_pel_backward_vector	1	
marker_bit	1	1
backward_f_code	3	
marker_bit	1	1
picture_coding_extension		
extension_start_code	32	000001B5
extension_start_code_identifier	4	8
f_code[0][0]	4	
f_code[0][1]	4	
f_code[1][0]	4	
f_code[1][1]	4	
intra_dc_precision	2	
picture_structure	2	
top_field_first	1	
frame_pred_frame_dct	1	
concealment_motion_vectors	1	
q_scale_type	1	
marker_bit	1	1
intra_vlc_format	1	
alternate_scan	1	
repeat_first_field	1	
chroma_420_type	1	
progressive_frame	1	
composite_display_flag	1	
v_axis	1	
field_sequence	3	
sub_carrier	1	
burst_amplitude	7	

第42図

history stream(40-4)

	bits	value
marker_bits	1	1
sub_carrier_phase	8	
quant_matrix_extension		
extension_start_code	32	000001B5
extension_start_code_identifier	4	3
quant_matrix_extension_present_flag	1	
load_intra_quantiser_matrix	1	
marker_bits	2	3
intra_quantiser_matrix[64]	8*64	
load_non_intra_quantiser_matrix	1	
marker_bits	7	7F
non_intra_quantiser_matrix[64]	8*64	
load_chroma_intra_quantiser_matrix	1	
marker_bits	7	7F
chroma_intra_quantiser_matrix[64]	8*64	
load_chroma_non_intra_quantiser_matrix	1	
marker_bits	7	7F
chroma_non_intra_quantiser_matrix[64]	8*64	
copyright_extension		
extension_start_code	32	000001B5
extension_start_code_identifier	4	4
copyright_extension_present_flag	1	
copyright_flag	1	
copyright_identifier	8	
original_or_copy	1	
marker_bit	1	
copyright_number_1	20	
marker_bit	1	
copyright_number_2	22	
marker_bit	1	
copyright_number_3	22	3F
marker_bits	6	

第43図

history stream(40-5)

	bits	value
picture_display_extension		
extension_start_code	32	000001B5
extension_start_code_identifier	4	7
picture_display_extension_present_flag	1	
frame_centre_horizontal_offset_1	16	
marker_bit	1	1
frame_centre_vertical_offset_1	16	
marker_bit	1	1
frame_centre_horizontal_offset_2	16	
marker_bit	1	1
frame_centre_vertical_offset_2	16	
marker_bit	1	1
frame_centre_horizontal_offset_3	16	
marker_bit	1	1
frame_centre_vertical_offset_3	16	
marker_bit	6	3F
re_coding_stream_information		
user_data_start_code	32	000001B2
re_coding_stream_info_ID	16	91EC
red_bw_flag	1	
red_bw_indicator	2	
marker_bit	5	1F
user_data		
user_data_start_code	32	000001B2
user_data	2048	
while(macroblock i=macroblock_count){		
macroblock		
macroblock_address_h	8	
macroblock_address_v	8	
slice_header_present_flag	1	
skipped_macroblock_flag	1	
marker_bit	1	1
macroblock_modes()		
macroblock_quant	1	
macroblock_motion_forward	1	
macroblock_motion_backward	1	
macroblock_pattern	1	
macroblock_intra	1	

第44図

history stream(40-6)

	bits	value
spatial_temporal_weight_code_flag	1	
frame_motion_type	2	
field_motion_type	2	
dct_type	1	
marker_bits	2	3
quantiser_scale_code	5	
marker_bits	3	7
PMV[0][0][0]	14	
marker_bits	2	3
PMV[0][0][1]	14	
motion_vertical_field_select[0][0]	1	
marker_bit	1	1
PMV[0][1][0]	14	
marker_bits	2	3
PMV[0][1][1]	14	
motion_vertical_field_select[0][1]	1	
marker_bit	1	1
PMV[1][0][0]	14	
marker_bits	2	3
PMV[1][0][1]	14	
motion_vertical_field_select[1][0]	1	
marker_bit	1	1
PMV[1][1][0]	14	
marker_bits	2	3
PMV[1][1][1]	14	
motion_vertical_field_select[1][1]	1	
marker_bit	1	1
coded_block_pattern	12	
marker_bits	4	F
num_mv_bits	8	
num_coef_bits	14	
marker_bits	2	3

第45図

history stream(40-7)

	bits	value
num_other_bits	7	
marker_bit	1	1
}		

第46図

history_stream(){	No. of bits	Mnemonic
next_start_code ()		
sequence_header ()		
sequence_extension ()		
extension_and_user_data (0)		
if(nextbits() == group_start_code){		
group_of_pictures_header()		
extension_and_user_data (1)		
}		
picture_header()		
picture_coding_extension()		
re_coding_stream_info()		
extensions_and_user_data (2)		
picture_data()		
sequence_end_code	32	bslbf
}		

第47図

sequence_header(){	No. of bits	Mnemonic
sequence_header_code ()	32	bslbf
horizontal_size_value	12	uimsbf
vertical_size_value	12	uimsbf
aspect_ratio_information	4	uimsbf
frame_rate_code	4	uimsbf
bit_rate_value	18	uimsbf
marker_bit	1	bslbf
vbv_buffer_size_value	10	uimsbf
constrained_parameters_flag	1	bslbf
load_intra_quantiser_matrix	1	uimsbf
if(load_intra_quantiser_matrix)		
intra_quantiser_matrix[64]	8*64	uimsbf
load_non_intra_quantiser_matrix	1	uimsbf
if(load_non_intra_quantiser_matrix)		
non_intra_quantiser_matrix[64]	8*64	uimsbf
next_start_code ()		
}		

第48図

sequence_extension(){	No. of bits	Mnemonic
extension_start_code	32	bslbf
extension_start_code_identifier	4	uimsbf
profile_and_level_indication	8	uimsbf
progressive_sequence	1	uimsbf
chroma_format	2	uimsbf
horizontal_size_extension	2	uimsbf
vertical_size_extension	2	uimsbf
bit_rate_extension	12	uimsbf
marker_bit	1	bslbf
vbv_buffer_size_extension	8	uimsbf
low_delay	1	uimsbf
frame_rate_extension_n	2	uimsbf
frame_rate_extension_d	5	uimsbf
next_start_code ()		
}		

第49図

extension_and_user_data (i){	No. of bits	Mnemonic
while((nextbits() == extension_start_code) 		
(nextbits() == user_data_start_code)){		
if((i == 2) && (nextbits() == extension_start_code))		
extension_data()		
if(nextbits() == user_data_start_code)		
user_data()		
}		
}		

第50図

user_data (){	No. of bits	Mnemonic
user_data_start_code	32	bslbf
while(nextbits() != '0000 0000 0000 0000 0000 0001'){		
user_data	8	uimsbf
}		
next_start_code()		
}		

第51図

group_of_pictures_header() {	No. of bits	Mnemonic
group_start_code	32	bslbf
time_code	25	bslbf
closed_gop	1	uimsbf
broken_link	1	uimsbf
next_start_code()		
}		

第52図

picture_header() {	No. of bits	Mnemonic
picture_start_code	32	bslbf
temporal_reference	10	uimsbf
picture_coding_type	3	uimsbf
vbv_delay	16	uimsbf
if(picture_coding_type==2 picture_coding_type==3){		
full_pel_forward_vector	1	bslbf
forward_f_code	3	bslbf
}		
if(picture_coding_type==3){		
full_pel_forward_vector	1	bslbf
backward_f_code	3	bslbf
}		
while(nextbits()=='1'){		
extra_bit_picture/*with the value '1'*/	1	uimsbf
extra_information_picture	8	uimsbf
}		
extra_bit_picture/*with the value '0'*/	1	uimsbf
next_start_code()		
}		

第53図

picture_coding_extension(){	No. of bits	Mnemonic
extension_start_code	32	bslbf
extension_start_code_idenfier	4	uimsbf
f_code [0] [0] /*forward horizontal*/	4	uimsbf
f_code [0] [1] /*forward vertical*/	4	uimsbf
f_code [1] [0] /*backward horizontal*/	4	uimsbf
f_code [1] [1] /*backward vertical*/	4	uimsbf
intra_dc_precision	2	uimsbf
picture_structure	2	uimsbf
top_field_first	1	uimsbf
frame_pred_frame_dct	1	uimsbf
concealment_motion_vectors	1	uimsbf
q_scale_type	1	uimsbf
intra_vlc_format	1	uimsbf
alternate_scan	1	uimsbf
repeat_first_field	1	uimsbf
chroma_420_type	1	uimsbf
progressive_frame	1	uimsbf
composite_display_flag	1	uimsbf
if(composite_display_flag){		
v_axis	1	uimsbf
field_sequence	3	uimsbf
sub_carrier	1	uimsbf
burst_amplitude	7	uimsbf
sub_carrier_phase	8	uimsbf
}		
next_start_code()		
}		

第54図

extension_data(){	No. of bits	Mnemonic
while(nextbits()==extension_start_code) {		
extension_start_code	32	bslbf
if (nextbits()=="Quant Matrix Extension ID")		
quant_matrix_extension()		
else if (nextbits()=="Copyright Extension ID")		
copyright_extension()		
else		
picture_display_extension()		
}		
}		

第55図

quant_matrix_extension(){	No. of bits	Mnemonic
extension_start_code_identifier	4	uimsbf
load_intra_quantiser_matrix	1	uimsbf
if (load_intra_quantiser_matrix)		
intra_quantiser_matrix [64]	8*64	uimsbf
load_non_intra_quantiser_matrix	1	uimsbf
if (load_non_intra_quantiser_matrix)		
non_intra_quantiser_matrix [64]	8*64	uimsbf
load_chroma_intra_quantiser_matrix	1	uimsbf
if (load_chroma_intra_quantiser_matrix)		
chroma_intra_quantiser_matrix [64]	8*64	uimsbf
load_chroma_non_intra_quantiser_matrix	1	uimsbf
if (load_chroma_non_intra_quantiser_matrix)		
chroma_non_intra_quantiser_matrix [64]	8*64	uimsbf
next_start_code()		
}		

第56図

copyright_extension() {	No. of bits	Mnemonic
extension_start_code_identifier	4	uimsbf
copyright_flag	1	bslbf
copyright_identifier	8	uimsbf
original_or_copy	1	bslbf
reserved	7	uimsbf
marker_bit	1	bslbf
copyright_number_1	20	uimsbf
marker_bit	1	bslbf
copyright_number_2	22	uimsbf
marker_bit	1	bslbf
copyright_number_3	22	uimsbf
next_start_code()		
}		

第57図

picture_display_extension() {	No. of bits	Mnemonic
extension_start_code_identifier	4	uimsbf
for(i=0;i:number_of_frame_centre_offsets;i++) {		
frame_centre_horizontal_offset	16	simsbf
marker_bit	1	bslbf
frame_centre_vertical_offset	16	simsbf
marker_bit	1	bslbf
}		
next_start_code()		
}		

第58図

picture_data() {	No. of bits	Mnemonic
while(nextbits()==slice_start_code) {		
slice()		
}		
next_start_code()		
}		

第59図

slice() {	No. of bits	Mnemonic
slice_start_code	32	bslbf
slice_quantiser_scale_code	5	uimsbf
if (nextbit()=='1') {		
intra_slice_flag	1	bslbf
intra_slice	1	uimsbf
reserved_bits	7	uimsbf
while(nextbits()=='1') {		
extra_bit_slice/*with the value '1'*/	1	uimsbf
extra_information_slice	8	uimsbf
}		
}		
extra_bit_slice/*with the value '0'*/	1	uimsbf
do {		
macroblock()		
} while(nextbit()!='000 0000 0000 0000 0000 0000')		
next_start_code()		
}		

第60図

macroblock() {	No. of bits	Mnemonic
while(nextbits()=='0000 0001 000')		
macroblock_escape	11	bslbf
macroblock_address_increment	1-11	vlclbf
macroblock_modes()		
if (macroblock_quant)		
macroblock_quantiser_scale_code	5	uimbsf
if (macroblock_motion_forward 		
(macroblock_intra && concealment_motion_vectors))		
motion_vectors(0)		
if (macroblock_motion_backward)		
motion_vectors(1)		
if (macroblock_intra && concealment_motion_vectors)		
marker_bit	1	bslbf
}		
}		

第61図

macroblock_modes() {	No. of bits	Mnemonic
macroblock_type	1-9	vlc1bf
if (macroblock_motion_forward 		
macroblock_motion_backward) {		
if (picture_structure=='frame') {		
if (frame_pred_frame_dct==0)		
frame_motion_type	2	uimsbf
} else {		
field_motion_type	2	uimsbf
}		
}		
if ((picture_structure=="Frame picture") &&		
(frame_pred_frame_dct==0) &&		
(dct_type_flag==1)) {		
dct_type	1	uimsbf
}		
}		

第62図

motion_vectors(s) {	No. of bits	Mnemonic
if (motion_vector_count==1) {		
if ((mv_format==field)&&(dmv!=1)		
motion_vertical_field_select [0][s]	1	uimsbf
motion_vector(0,s)		
} else {		
motion_vertical_field_select [0][s]	1	uimsbf
motion_vector(0,s)		
motion_vertical_field_select [1][s]	1	uimsbf
motion_vector(1,s)		
}		
}		

第63図

motion_vectors(r,s) {	No. of bits	Mnemonic
motion_code[r][s][0]	1-11	vlcibf
if ((f_code[s][0]!=1)&&(motion_code[r][s][0]!=0)		
motion_residual [r][s][0]	1-8	uimsbf
if (dmv==1)		
dmvector [0]	1-2	vlcibf
motion_code[r][s][1]	1-11	vlcibf
if ((f_code[s][1]!=1)&&(motion_code[r][s][1]!=0)		
motion_residual [r][s][1]	1-8	uimsbf
if (dmv==1)		
dmvector [1]	1-2	vlcibf
}		

第64図

macroblock_type V L C code					
		macroblock_quant			
				dct_type_flag	
				macroblock_motion_forward	
				macroblock_motion_backward	
				Description	
1	0	1	0	0	Intra
01	1	1	0	0	Intra, Quant

第65図

macroblock_type V L C code						
		macroblock_quant				
					dct_type_flag	
					macroblock_motion_forward	
					macroblock_motion_backward	
					Description	
1		0	1	1	0	MC, Coded
01		0	1	0	0	No MC, Coded
001		0	0	0	0	MC, Not Coded
0001	1	0	1	0	0	Intra
0001	0	1	1	1	0	MC, Coded, Quant
0000	1	1	1	0	0	No MC, Coded, Quant
0000	01	1	1	0	0	Intra, Quant

第66図

macroblock_type V L C code					
		macroblock_quant			
		dct_type_flag			
		macroblock_motion_forward			
		macroblock_motion_backward			
		Description			
10	0	0	0	0	Interp, Not Coded
11	0	1	1	1	Interp, Coded
010	0	0	0	0	Bwd, Not Coded
011	0	1	0	1	Bwd, Coded
0010	0	0	0	0	Fwd, Not Coded
0011	0	1	1	0	Fwd, Coded
0001 1	0	1	0	0	Intra
0001 0	1	1	1	1	Interp, Coded, Quant
0000 11	1	1	1	0	Fwd, Coded, Quant
0000 10	1	1	0	1	Bwd, Coded, Quant
0000 01	1	1	0	0	Intra, Quant

第67図

re_coding_stream_info () {	No. of bits	Mnemonic
user_data_start_code	32	bslbf
re_coding_stream_info_ID	16	bslbf
red_bw_flag	1	uimsbf
if (red_bw_flag)		
red_bw_indicator	2	uimsbf
if (!red_bw_flag) {		
for(i=0;i<number_of_macroblock:++) {		
marker_bit	3	bslbf
num_other_bits	7	uimsbf
num_mv_bits	8	uimsbf
num_coef_bits	14	uimsbf
}		
}		
next_start_code()		
}		

第68図

Data Set	red_bw_flag	
		red_bw_indicator
Data Set 1	0	—
Data Set 2	1	0
Data Set 3	1	1
Data Set 4	1	2
Data Set 5	1	3

第69図

Data Set	num_coef_bits, num_mv_bits, num_other_bits										
	q_scale_code, q_scale_type										
	motion_type, mv_vert_field_sel[], mv[][]										
	mb_mfwd, mb_mbwd										
	mb_pattern										
	coded_block_pattern										
	mb_intra										
	slice_start										
	dct_type										
	mb_quant										
	skipped_mb										
Data Set 1	2	2	2	2	2	2	2	2	2	2	
Data Set 2	0	2	2	2	2	2	2	2	2	2	
Data Set 3	0	2	2	2	2	0	2	1	2	1	
Data Set 4	0	2	0	1	1	0	1	1	0	1	
Data Set 5	0	0	0	0	0	0	0	0	0	0	

履歴情報の項目の組合せ

第70図

picture rate elements(72-1)

parameter	number format	number of bits	bit offset from	bit offset to	data category	details
MPEG standard flag	1bit flag	1	0	0	3	1=>MPEG 1.0=>MPEG2
red_bw_flag	1bit flag	1	1	1	3	Default="0"
red_bw_indicator	2bit ui	3	2	4	3	Default="000"
header present flags	2bit flags	2	5	6	3	sequence header present flag, GOP header present flag
Extension start code flags	16 flags	16	7	22	3	Indicates if a given extension start code exists. The 16 flags correspond to the 16 entries in table 6.2 of the ISO/IEC 13818-2: 1996 standard in the order they are listed.
Other start codes	3 flags	3	23	25	3	user_data_start_code, sequence_error_code, sequence_end_code
sequence header						
horizontal_size	14bit uimsbf	14	26	39	2	includes extension
vertical_size	14bit uimsbf	14	40	53	2	includes extension
aspect_ratio_information	4bit uimsbf	4	54	57	2	
frame_rate_code	4bit uimsbf	4	58	61	2	
bit_rate	30bit uimsbf	30	62	91	2	includes extension, correct value should be calculated
vbv_buffer_size	18bit uimsbf	18	92	109	2	includes extension
constrained_parameters_flag	1bit flag	1	110	110	2	
sequence extension						
profile_and_level_indication	8bit uimsbf	8	111	118	2	
progressive_sequence	1bit flag	1	119	119	2	
chroma_format	2bit uimsbf	2	120	121	2	
low_delay	1bit flag	1	122	122	2	
sequence display extension						
video_format	3bit uimsbf	3	123	125	2	
colour_description	1bit flag	1	126	126	2	
colour_primaries	8bit uimsbf	8	127	134	2	
transfer_characteristics	8bit uimsbf	8	135	142	2	
matrix_coefficients	8bit uimsbf	8	143	150	2	
display_horizontal_size	14bit uimsbf	14	151	164	2	
display_vertical_size	14bit uimsbf	14	165	178	2	
group of pictures header						
time_code	25bit flag	25	179	203	2	
closed_gop	1bit flag	1	204	204	2	

第72図

picture rate elements(72-2)

broken_link	1bit flag	1	205	205	2	
picture header						
temporal_reference	10bit uimsbf	10	206	215	1	
picture_coding_type	3bit uimsbf	3	216	218	1	
vbv_delay	16bit uimsbf	16	219	234	1	should be calculated if not present in bitstream
full_pel_forward_vector	1bit flag	1	235	235	1	
forward_f_code	3bit uimsbf	3	236	238	1	
full_pel_backward_vector	1bit flag	1	239	239	1	
backward_f_code	3bit uimsbf	3	240	242	1	
picture coding extension						
forward_horizontal_f_code	4bit uimsbf	4	243	246	1	
forward_vertical_f_code	4bit uimsbf	4	247	250	1	
backward_horizontal_f_code	4bit uimsbf	4	251	254	1	
backward_vertical_f_code	4bit uimsbf	4	255	258	1	
intra_dc_precision	2bit uimsbf	2	259	260	1	
picture_structure	2bit uimsbf	2	261	262	1	
top_field_first	1bit flag	1	263	263	1	
frame_pred_frame_dct	1bit flag	1	264	264	1	
concealment_motion_vectors	1bit flag	1	265	265	1	
q_scale_type	1bit flag	1	266	266	1	
intra_vlc_format	1bit flag	1	267	267	1	
alternate_scan	1bit flag	1	268	268	1	
repeat_first_field	1bit flag	1	269	269	1	
chroma_420_type	1bit flag	1	270	270	1	
progressive_frame	1bit flag	1	271	271	1	
composite_display_flag	1bit flag	1	272	272	1	
v_axis	1bit flag	1	273	273	1	
field_sequence	3bit uimsbf	3	274	276	1	
sub_carrier	1bit flag	1	277	277	1	
burst_amplitude	7bit uimsbf	7	278	284	1	
sub_carrier_phase	8bit uimsbf	8	285	292	1	
quant_matrix_extension						
load_intra_quantiser_matrix	1bit flag	1	293	293	1	(1)
load_non_intra_quantiser_matrix	1bit flag	1	294	294	1	(1)
load_chroma_intra_quantiser_matrix	1bit flag	1	295	295	1	(1)
load_chroma_non_intra_quantiser_matrix	1bit flag	1	296	296	1	(1)
intra_quantiser_matrix[64]	64*0...255	512	297	808	2	(1)
non_intra_quantiser_matrix[64]	64*0...255	512	809	1320	2	(1)
chroma_intra_quantiser_matrix[64]	64*0...255	512	1321	1832	2	(1)
chroma_non_intra_quantiser_matrix[64]	64*0...255	512	1833	2344	2	(1)
picture_display_extension						
frame_centre_horizontal_offset_1	16bit uimsbf	16	2345	2360	2	
frame_centre_vertical_offset_1	16bit uimsbf	16	2361	2376	2	
frame_centre_horizontal_offset_2	16bit uimsbf	16	2377	2392	2	

第73图

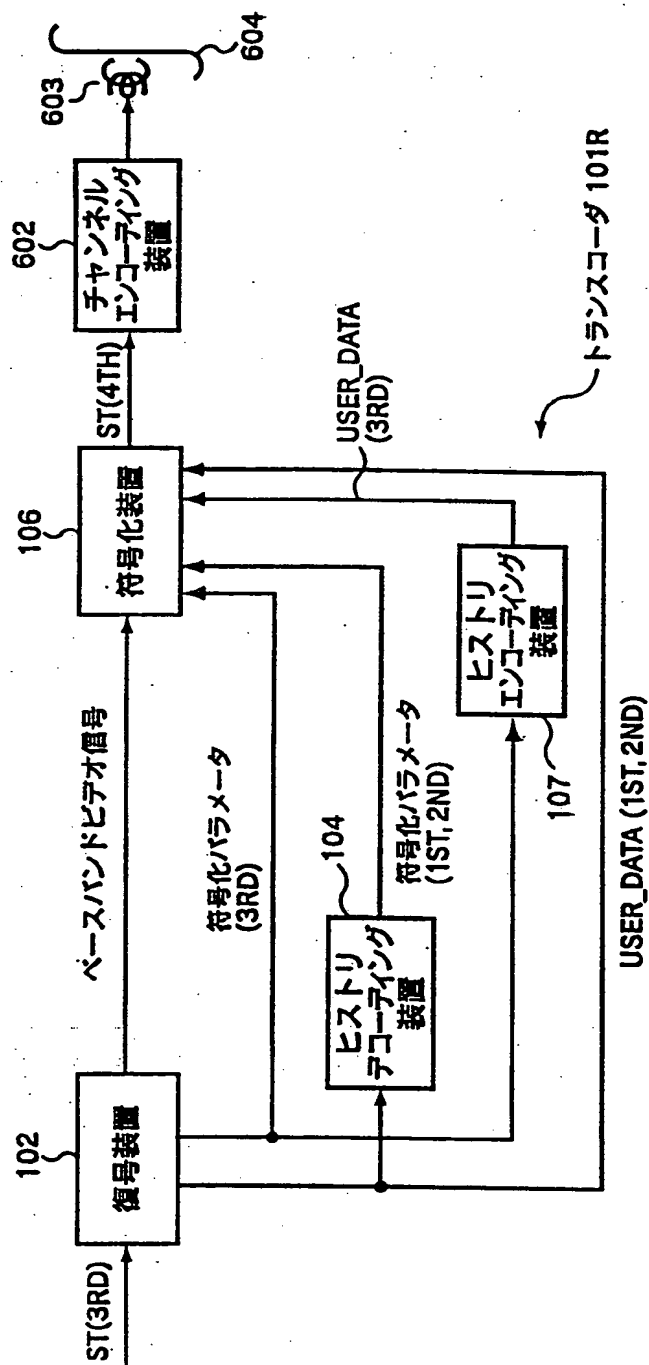
picture rate elements(72-3)

frame_centre_vertical_offset_2	16bit uimsbf	16	2393	2408	2	
frame_centre_horizontal_offset_3	16bit uimsbf	16	2409	2424	2	
frame_centre_vertical_offset_3	16bit uimsbf	16	2425	2440	2	
copyright_extension						
copyright_flag	1bit flag	1	2441	2441	2	
copyright_identifier	8bit code	8	2442	2449	2	
original_or_copy	1bit flag	1	2450	2450	2	
copyright_number	64bit uimsbf	64	2451	2514	2	
PTS/DTS						
PTS_DTS_flag	2bit flag	2	2515	2516	1	
PTS_Value	33bit uimsbf	33	2517	2549	2	
DTS_Value	33bit uimsbf	33	2550	2582	2	
spare reserved bits						
spare	41bit uimsbf	41	2583	2623		
user data area						
user_data		1664	2624	4287	2	
picture rate information CRC						
32-bit_protection_CRT	32bit uimsbf	32	4288	4319		

第74図

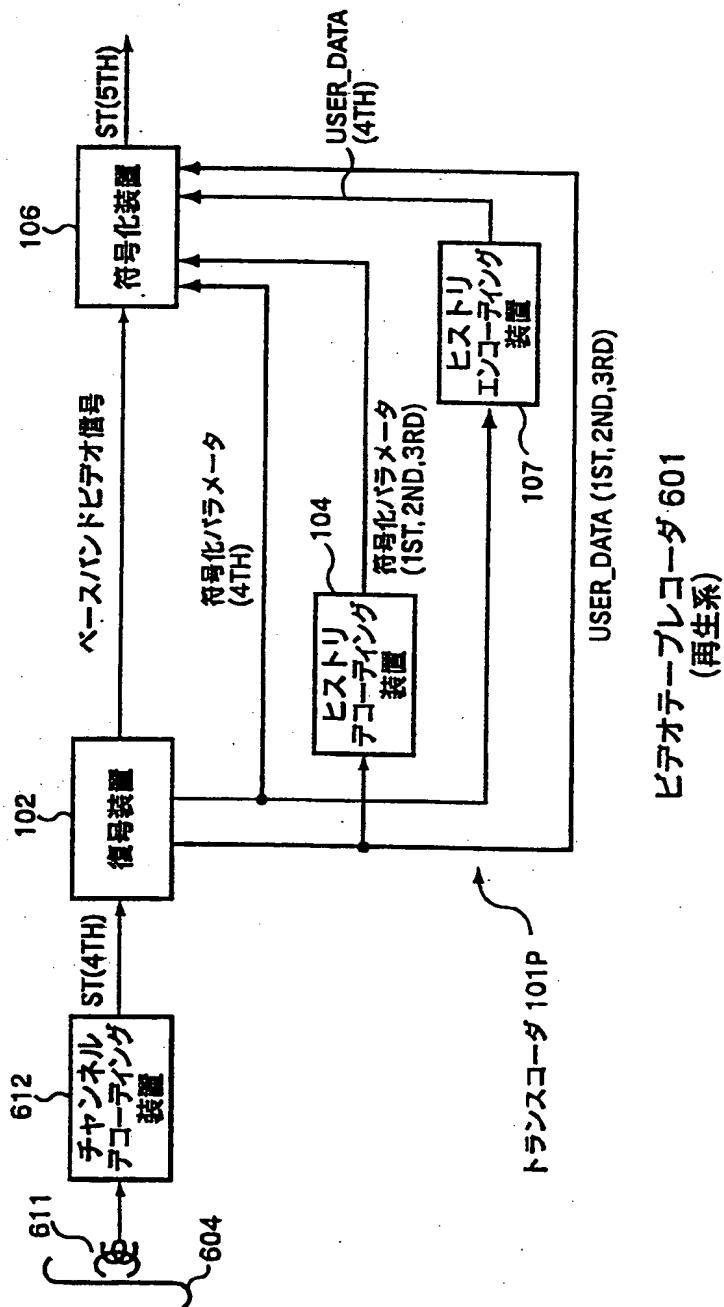
D9	Cb[0][9]	Y[0][9]	Cr[0][9]	Y[1][9]	Cb[1][9]	Y[2][9]	Cr[1][9]	Y[3][9]
D8	Cb[0][8]	Y[0][8]	Cr[0][8]	Y[1][8]	Cb[1][8]	Y[2][8]	Cr[1][8]	Y[3][8]
D7	Cb[0][7]	Y[0][7]	Cr[0][7]	Y[1][7]	Cb[1][7]	Y[2][7]	Cr[1][7]	Y[3][7]
D6	Cb[0][6]	Y[0][6]	Cr[0][6]	Y[1][6]	Cb[1][6]	Y[2][6]	Cr[1][6]	Y[3][6]
D5	Cb[0][5]	Y[0][5]	Cr[0][5]	Y[1][5]	Cb[1][5]	Y[2][5]	Cr[1][5]	Y[3][5]
D4	Cb[0][4]	Y[0][4]	Cr[0][4]	Y[1][4]	Cb[1][4]	Y[2][4]	Cr[1][4]	Y[3][4]
D3	Cb[0][3]	Y[0][3]	Cr[0][3]	Y[1][3]	Cb[1][3]	Y[2][3]	Cr[1][3]	Y[3][3]
D2	Cb[0][2]	Y[0][2]	Cr[0][2]	Y[1][2]	Cb[1][2]	Y[2][2]	Cr[1][2]	Y[3][2]
D1	Cb[0][1]	Y[0][1]	Cr[0][1]	Y[1][1]	Cb[1][1]	Y[2][1]	Cr[1][1]	Y[3][1]
D0	Embedded Aligned MPEG-2 Re-Coding Information Bus	Y[0][0]	Embedded Aligned MPEG-2 Re-Coding Information Bus	Y[1][0]	Embedded Aligned MPEG-2 Re-Coding Information Bus	Y[2][0]	Embedded Aligned MPEG-2 Re-Coding Information Bus	Y[3][0]

第75図

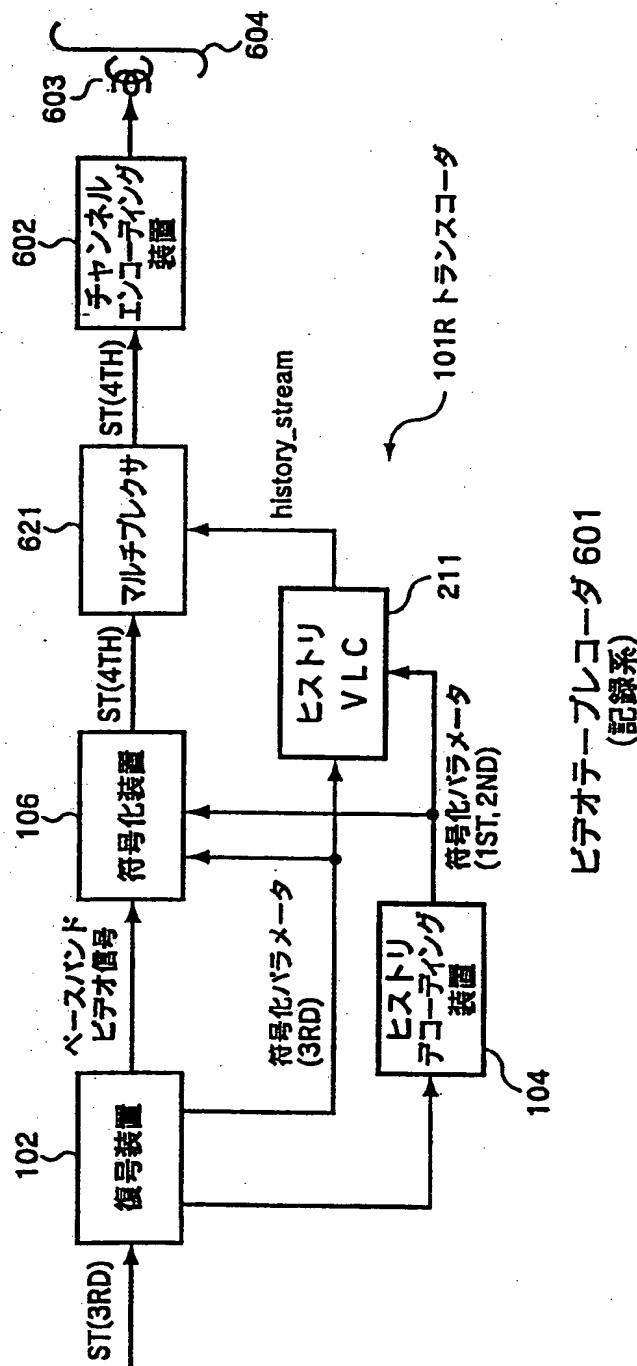


ビデオテーブルコーダ 601
(記録系)

第76図

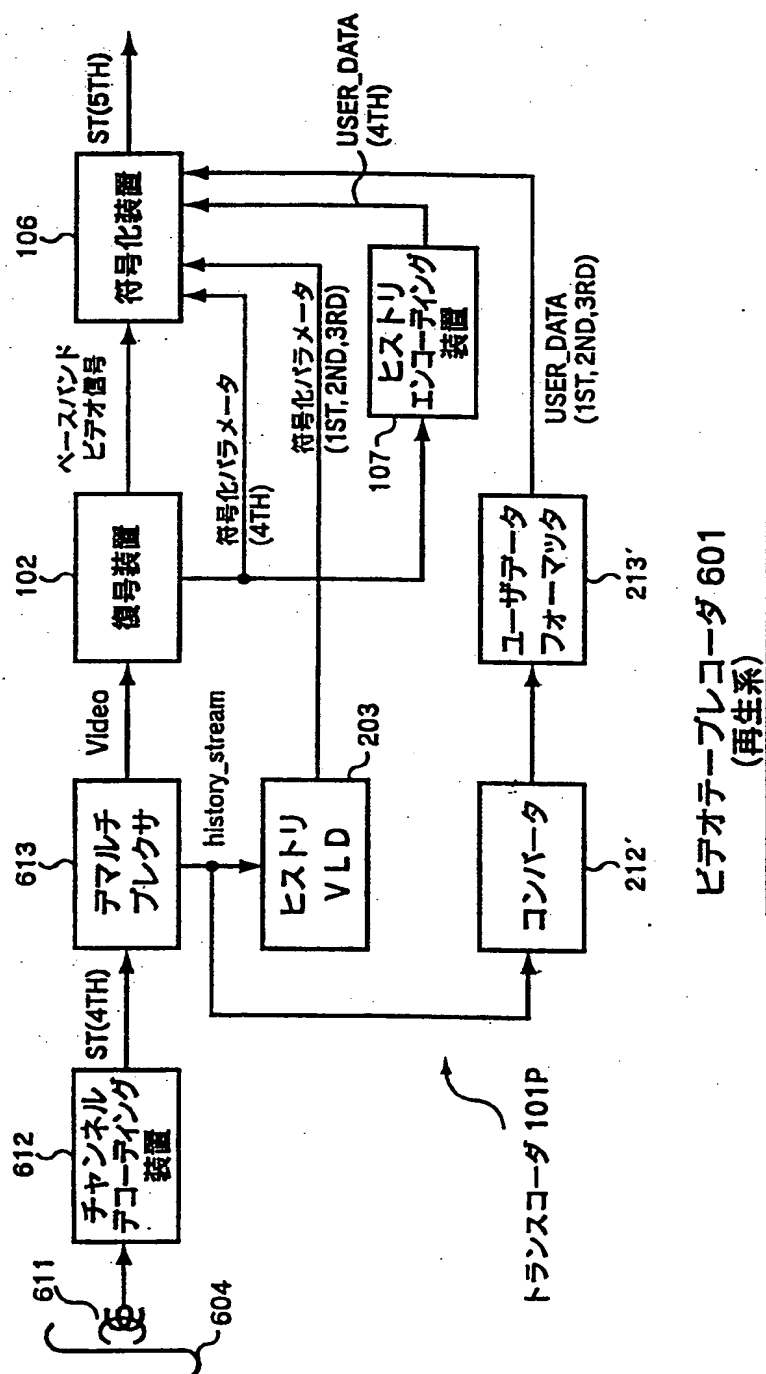


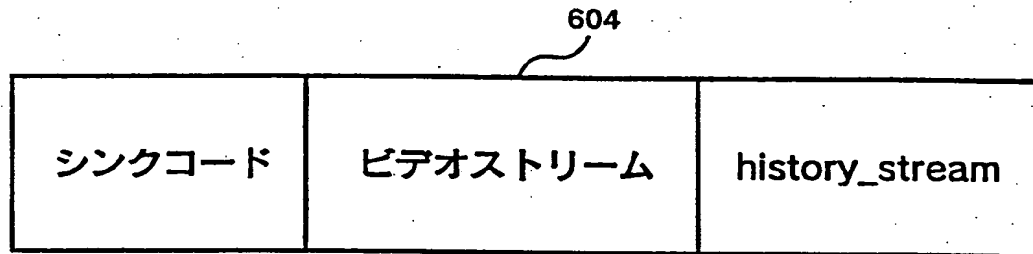
第77図



第78図

ビデオデコーダ 601
(記録系)





第80図

符 号 の 説 明

1、106……符号化装置、 2、102……復号装置、 3……記録媒体、 14、33……フレームメモリー、 17、32……フォーマット変換回路、 18……エンコーダ、 19……記録回路、 30……再生回路、 31……デコーダ、 50……動きベクトル検出回路、 57……量子化回路、 58……可変長符号化回路、 59……送信バッファ、 64、87……動き補償回路、 70……コントローラ、 81……受信バッファ、 82……可変長復号回路、 83……逆量子化回路、 84……IDCT回路、 100……コンピュータ、 101……トランスコーダ、 103……ヒストリ情報多重化装置、 104……ヒストリデコーディング装置、 105……ヒストリ情報分離装置、 107……ヒストリエンコーディング装置、 201……ユーザデータデコーダ、 202、212……コンバータ、 203、211……ヒストリVLC。

INTERNATIONAL SEARCH REPORT

International application No.

PCT/JP00/00720

A. CLASSIFICATION OF SUBJECT MATTER
Int.Cl.⁷ H04N7/50

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

Int.Cl.⁷ H04N7/24-7/68

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Jitsuyo Shinan Koho	1922-1996	Toroku Jitsuyo Shinan Koho	1994-2000
Kokai Jitsuyo Shinan Koho	1971-2000	Jitsuyo Shinan Toroku Koho	1996-2000

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	EP, 889650, A2 (Hewlett-Packard Company), 08 April, 1998 (08.04.98),	1-3, 24, 25, 42, 43, 48, 49, 52, 53
Y	Full text; Figs. 1 to 7	9-23, 28-41, 44, 45, 50, 51
A	& JP, 11-74798, A Full text; Figs. 1 to 7	5-8, 26, 27, 46, 47, 54, 55
Y	GB, 2318246, A (Sony United Kingdom Company), 15 April, 1998 (15.04.98),	1-3, 9-23, 24, 25, 28-45, 48-53
A	Full text; Figs. 1 to 11 & JP, 10-136386, A Full text; Figs. 1 to 9	5-8, 26, 27, 46, 47, 54, 55
A	JP, 8-111870, A (Kokusai Denshin Denwa Co., Ltd. (KDD)), 30 April, 1996 (30.04.96), Full text; Figs. 1 to 4 (Family: none)	1-55
A	JP, 8-130743, A (Mitsubishi Electric Corporation), 21 May, 1996 (21.05.96), Full text; Figs. 1 to 5 & EP, 710030, A1 & US, 5831688, A	1-55
A	JP, 7-288804, A (Kokusai Denshin Denwa Co., Ltd. (KDD)),	1-55

☒ Further documents are listed in the continuation of Box C.☐ See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance
 "E" earlier document but published on or after the international filing date
 "L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
 "O" document referring to an oral disclosure, use, exhibition or other means
 "P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
 "X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
 "Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
 "&" document member of the same patent family

Date of the actual completion of the international search
09 May, 2000 (09.05.00)Date of mailing of the international search report
23 May, 2000 (23.05.00)Name and mailing address of the ISA/
Japanese Patent Office

Authorized officer

Facsimile No.

Telephone No.

INTERNATIONAL SEARCH REPORT

International application No.

PCT/JP00/00720**C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT**

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
	31 October, 1995 (31.10.95), Full text; Figs. 1 to 7 (Family: none)	

国際調査報告

国際出願番号 PCT/JP00/00720

A. 発明の属する分野の分類 (国際特許分類 (IPC))

Int. Cl⁷ H04N7/50

B. 調査を行った分野

調査を行った最小限資料 (国際特許分類 (IPC))

Int. Cl⁷ H04N7/24-7/68

最小限資料以外の資料で調査を行った分野に含まれるもの

日本国実用新案公報 1922-1996年

日本国公開実用新案公報 1971-2000年

日本国登録実用新案公報 1994-2000年

日本国実用新案登録公報 1996-2000年

国際調査で使用した電子データベース (データベースの名称、調査に使用した用語)

C. 関連すると認められる文献

引用文献の カテゴリー*	引用文献名 及び一部の箇所が関連するときは、その関連する箇所の表示	関連する 請求の範囲の番号
X	EP, 889650, A2 (Hewlett-Packard Company) 8. 4月. 1998 (08. 04. 98) 全頁, 第1-7図	1-3, 24, 25, 42, 43, 48, 49, 52, 53
Y	& JP, 11-74798, A 全頁, 第1-7図	9-23, 28-41, 44, 45, 50, 51
A		5-8, 26, 27, 46, 47, 54, 55

☒ C欄の続きにも文献が列挙されている。☐ パテントファミリーに関する別紙を参照。

* 引用文献のカテゴリー

「A」 特に関連のある文献ではなく、一般的技術水準を示すもの

「E」 国際出願日前の出願または特許であるが、国際出願日以後に公表されたもの

「L」 優先権主張に疑義を提起する文献又は他の文献の発行日若しくは他の特別な理由を確立するために引用する文献 (理由を付す)

「O」 口頭による開示、使用、展示等に言及する文献

「P」 国際出願日前で、かつ優先権の主張の基礎となる出願

の日の後に公表された文献

「T」 国際出願日又は優先日後に公表された文献であって出願と矛盾するものではなく、発明の原理又は理論の理解のために引用するもの

「X」 特に関連のある文献であって、当該文献のみで発明の新規性又は進歩性がないと考えられるもの

「Y」 特に関連のある文献であって、当該文献と他の1以上の文献との、当業者にとって自明である組合せによって進歩性がないと考えられるもの

「&」 同一パテントファミリー文献

国際調査を完了した日 09. 05. 00

国際調査報告の発送日 23.05.00

国際調査機関の名称及びあて先
 日本国特許庁 (ISA/JP)
 郵便番号100-8915
 東京都千代田区霞が関三丁目4番3号

特許庁審査官 (権限のある職員)
 藤内 光武 印 5P 9746

電話番号 03-3581-1101 内線 3581

C (続き). 関連すると認められる文献

引用文献の カテゴリー*	引用文献名 及び一部の箇所が関連するときは、その関連する箇所の表示	関連する 請求の範囲の番号
Y	GB, 2318246, A (Sony United Kingdom Company) 15. 4月. 1998 (15. 04. 98) 全頁, 第1-11図 & JP, 10-136386, A 全頁, 第1-9図	1-3, 9-23, 24, 25, 28-45, 48-53 5-8, 26, 27, 46, 47, 54, 55
A		
A	JP, 8-111870, A (国際電信電話株式会社) 30. 4月. 1996 (30. 04. 96) 全頁, 第1-4図 (ファミリーなし)	1-55
A	JP, 8-130743, A (三菱電機株式会社) 21. 5月. 1996 (21. 05. 96) 全頁, 第1-5図 & EP, 710030, A1 & US, 5831688, A	1-55
A	JP, 7-288804, A (国際電信電話株式会社) 31. 10月. 1995 (31. 10. 95) 全頁, 第1-7図 (ファミリーなし)	1-55

**This Page is Inserted by IFW Indexing and Scanning
Operations and is not part of the Official Record**

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ **BLACK BORDERS**
- ☐ **IMAGE CUT OFF AT TOP, BOTTOM OR SIDES**
- ☐ **FADED TEXT OR DRAWING**
- ☐ **BLURRED OR ILLEGIBLE TEXT OR DRAWING**
- ☐ **SKEWED/SLANTED IMAGES**
- ☐ **COLOR OR BLACK AND WHITE PHOTOGRAPHS**
- ☐ **GRAY SCALE DOCUMENTS**
- ☐ **LINES OR MARKS ON ORIGINAL DOCUMENT**
- ☐ **REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY**
- ☐ **OTHER:** _____

IMAGES ARE BEST AVAILABLE COPY.

As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.